STUDY ON NEUTRON ENERGY SPECTRUM CORRECTION AND PULSE SHAPE DISCRIMINATION WITH A BORON-LOADED SCINTILLATOR

by

Can Liao

A thesis submitted to the faculty of
The University of Utah
in partial fulfillment of the requirements for the degree of

Master of Science

in

Nuclear Engineering

Department of Civil and Environmental Engineering

The University of Utah

December 2012

The University of Utah Graduate School

STATEMENT OF THESIS APPROVAL

The thesis of **Can Liao**

has been approved by the following supervisory committee members:

Haori Yang	, Chair	**10/19/2012** Date Approved
Tatjana Jevremovic	, Member	**10/19/2012** Date Approved
Dong-Ok Choe	, Member	**10/19/2012** Date Approved

and by **Chris Pantelides** , Chair of

the Department of **Civil and Environmental Engineering**

and by Charles A. Wight, Dean of The Graduate School.

ABSTRACT

In this thesis, the fast neutron spectrometry and pulse shape discrimination feature were studied using a Boron-loaded liquid scintillator, which utilizes the capture-gated counting method. This method is based on the measurement of two signals generated by the sequential interactions from the same neutron in the detector. The neutron-capture pulse is gated by the recoil-proton pulse so that only neutrons whose energy has been finally absorbed are recorded, because the amplitude of the recoil-proton pulse does not reflect the energy of the incident neutron, unless the neutron is successfully thermalized and captured. When a neutron fully deposits its energy in the active region of the detector, the recoil-proton pulse amplitude is strongly correlated to the incident neutron energy. This serves as the basis of neutron spectroscopy.

In current practice of the capture-gated technique, it is always assumed that capture only happens after the incoming neutron loses all of its kinetic energy. In this thesis, we show that this assumption may cause degradation of energy resolution that can be corrected based on capture time. The concept of our approach is demonstrated by simulation using the Geant4 toolkit. In addition, the pulse shape discrimination feature of this detector will be discussed using Geant4 simulation as well.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

This dissertation would not have been possible without the guidance and the help of several individuals who contributed and extended their valuable assistance in the preparation and completion of my research.

First and foremost, my advisor Dr. Yang, assistant professor of the Nuclear Engineering Program, University of Utah, has taught me basic research methods and provided much help with my difficulties in the completion of this research work. He also inspired me a lot when I lacked courage and motivation.

Dr. Jevremovic, professor and director of the Nuclear Engineering Program, University of Utah, agreed to be my committee member and provide me a lot of help on research work.

Dr. Choe, assistant research professor and TRIGA reactor supervisor of the Nuclear Engineering Program, University of Utah, has provided me much assistance when doing experiments in the facility.

The administrators and faculty of the Civil and Environmental Engineering Department, University of Utah, and my colleagues in the Nuclear Engineering Program have also helped a lot.

Last but not the least comes my family. They are always there for me when I'm down and have encouraged me to chase my own life.

CHAPTER 1

INTRODUCTION

1.1 Neutron Spectrometry

Accurate neutron energy measurements are required in many research areas, such as nuclear safeguards [1], plasma diagnostics [2], and radiation therapy [3]. Neutron spectrometry is not as easy as energy spectrometry for other ionizing radiations such as alpha and gamma rays, because neutrons cannot directly ionize materials and consequently have to be detected by transferring their energy to other ionizing particles via scattering or other types of reactions. Furthermore, it is difficult to find a detection mechanism that works for a very wide range of energies because different kinds of reactions dominate in different energy ranges.

Many techniques have been developed and studied for several decades. Spectrometry based on time-of-flight, nuclear reaction, and nuclear recoil are existing techniques that have been widely used in many applications.

1.2 Boron-loaded Liquid Scintillator Operated in Capture-gated Mode

Scintillation detectors can be used to detect almost all types of radiation. A typical scintillation detector is constituted of a scintillator, a photoelectric conversion device, and other involved electronics. Scintillators can be divided into organic and inorganic

scintillators according to their constitutions. They can also be categorized as liquid, solid, and gaseous scintillators. A photomultiplier tube (PMT) is the most widely used photoelectric conversion device. When an ionizing particle enters the scintillator, atoms or molecules are excited and then de-excite by emitting optical photons, which is called fluorescence. These optical photons are captured by a PMT after being collected by reflective material or a light guide, generating photoelectrons via the photoelectric effect. After the subsequent multiplication, these photoelectrons form an electric pulse whose charge is proportional to the energy deposition by the incident particle.

As mentioned in Section 1.1, neutrons are not ionizing particles; thus, they cannot generate fluorescence directly. In a regular organic scintillator, neutrons are mostly scattered by hydrogen nuclei generating recoil protons that could be measured to estimate the incident neutron energy deposition. However, neutrons can escape before they lose all of their energy; therefore, the neutron energy estimated through recoil protons could be inaccurate, which leads to poor energy resolution. This fact motivated researchers to load neutron-absorbing materials into the regular organic scintillator to maximize thermal neutron capture.

The capture-gated method using a loaded liquid scintillator is a robust and reliable method for fast neutron spectrometry [4], [5]. This technique is based on the measurement of two subsequent signals generated by the same neutron interacting with the scintillator. An incident fast neutron is not easily captured because the absorption cross-section is small in the fast range. It will be scattered multiple times, mainly by hydrogen nuclei and generate recoil protons. After the neutron is thermalized, it has a very high probability of being captured by the neutron-absorbing medium, such as ^{10}B,

^{6}Li, and ^{113}Cd. If the neutron is captured, two pulses will be produced: one by recoils protons and another by absorption of the thermalized neutron. The prompt recoil-proton pulse is gated by the capture pulse generated by the same neutron. The primary (scattering) neutron pulse from the scintillator is accepted as the 'true' neutron pulse only if the subsequent (capture) neutron pulse is detected. The prompt pulse amplitude is strongly correlated to the incident neutron energy; thus, this method can be reliably used to estimate the incident neutron energy [6-8].

1.3 Pulse Shape Discrimination

Boron-10-loaded liquid scintillators offer high sensitivity to both fast and thermal neutrons as well as to γ–rays. However, this sensitivity to various types of radiation can be considered as an advantage only if the events generated by different types of radiation could be easily and accurately separated. In a liquid scintillator working in capture-gated mode, there will be three kinds of pulses generated by the recoil protons, and the capture of neutron and gamma particles, respectively. Thus, an effective algorithm to discriminate the recoil-proton pulses, the neutron-capture pulses and gamma ray pulses is quite essential to achieve accurate spectrum reconstruction. Pulse Shape Discrimination (PSD) is the basic method that we used to distinguish these pulses.

1.4 Thesis Outline

In this thesis, the accuracy of spectrum derived from a boron-loaded liquid scintillator operated in capture-gated mode will be analyzed. A method to correct the inaccuracy in energy measurement has been developed. Simulations and experiments are both

performed to demonstrate the theory and analysis process. A boron-loaded liquid scintillation detector (EJ-339A, 4.6% ^{10}B atom percent) has been chosen as an example scintillation detector.

CHAPTER 2

THEORY AND METHOD

2.1 Interactions of Neutrons with Materials

Neutrons are usually categorized according to their energy, because different types of neutron interaction dominate in different neutron energy ranges. The most common classification is shown in Table 2.1 [9].

2.1.1 Types of Neutron Interaction

There are two major types interactions of neutron with nuclei: scattering and absorption.

In an absorption interaction, both the nucleus and the incident neutron disappear. One or more other particles are produced after the interaction. Absorption is divided into several kinds of reactions according to the process and particles created after the absorption reaction. Radioactive capture (n, γ) is a neutron capture process whose prompt result is the emission of electromagnetic radiation only [10]. In this process, nucleus ${}^{A}_{Z}X$ becomes ${}^{A+1}_{Z}X$ after absorbing a neutron and emitting a gamma ray. In a Charged Particle Emission interaction, charged particles, such as alpha particles (n, α) or protons (n, p), are emitted after the absorption. Fission (n, f) is a nuclear reaction in which the nucleus of an

Table 2.1 Classifications of Neutrons

Neutron energy	Name
0-0.025 eV	Cold
0.025 eV	Thermal
0.025 eV-0.4 eV	Epithermal
0.4 eV-0.6 eV	Cadmium
0.6 eV-1 eV	Epicadmium
1 eV-10 eV	Slow
10 eV-300 eV	Resonance
300 eV-1 MeV	Intermediate
1MeV-20 MeV	Fast
>20MeV	Relativistic

atom splits into smaller parts (lighter nuclei) after absorbing a neutron. During a fission reaction, free neutrons and gamma rays are often produced as well.

2.1.2 Neutron Reaction Cross-Section

The neutron reaction cross-section is used to characterize the probability that a neutron reaction will occur. Microscopic cross-section and macroscopic cross-section are two different ways to describe the probability in different magnitudes.

In order to introduce the concept of microscopic cross-section, consider a parallel monoenergetic neutron beam incident on a target perpendicularly. The thickness of the target is t and the number density of the target material is N. The intensity of the incident neutron beam is $I_0=n\nu$, where n is the number of neutrons per unit volume and ν is the neutron velocity. In such an experiment, the number of neutrons that interact with the target nuclei is proportional to the neutron beam intensity and number of target nuclei:

$$N_{reaction} = \sigma I_0 Nat \quad \text{Eq. 2.1}$$

where a is the area of target struck by the beam and $N_{reaction}$ is the number of neutrons that interact with the target nuclei.

The parameter σ, the proportionality coefficient, is called the microscopic cross-section. Thus, the microscopic cross-section has the following physical meaning: the probability that an interaction will occur per target nucleus per neutron per unit area. The unit of σ is barn (b). 1 b = 10^{-24} cm^2. Another form of the cross-section, the macroscopic cross-section Σ, is defined by the following equation:

$$\Sigma = N\sigma \quad \text{Eq. 2.2}$$

where, N is the number density of target nuclei.

It describes the probability that an interaction will take place per unit distance of travel for a neutron moving in a medium that has an atom density of N.

Neutron cross-sections are also defined respectively for each type of reaction discussed in Section 2.1 and for each isotope. For example, σ_s (^{10}B) describes the elastic scattering cross-section of boron-10 and σ_f (^{235}U) describes the fission cross-section of Uranium-235. Fig. 2.1 shows the total absorption microscopic cross-section of Uranium-235.

In Fig. 2.1, region 1 is called the '1/v' region (low energy). In this region, the cross-section value is inversely proportional to $\sqrt{E}$. Since energy is proportional to the square of the speed, the cross-section value is then inversely proportional to v. Region 2 is the

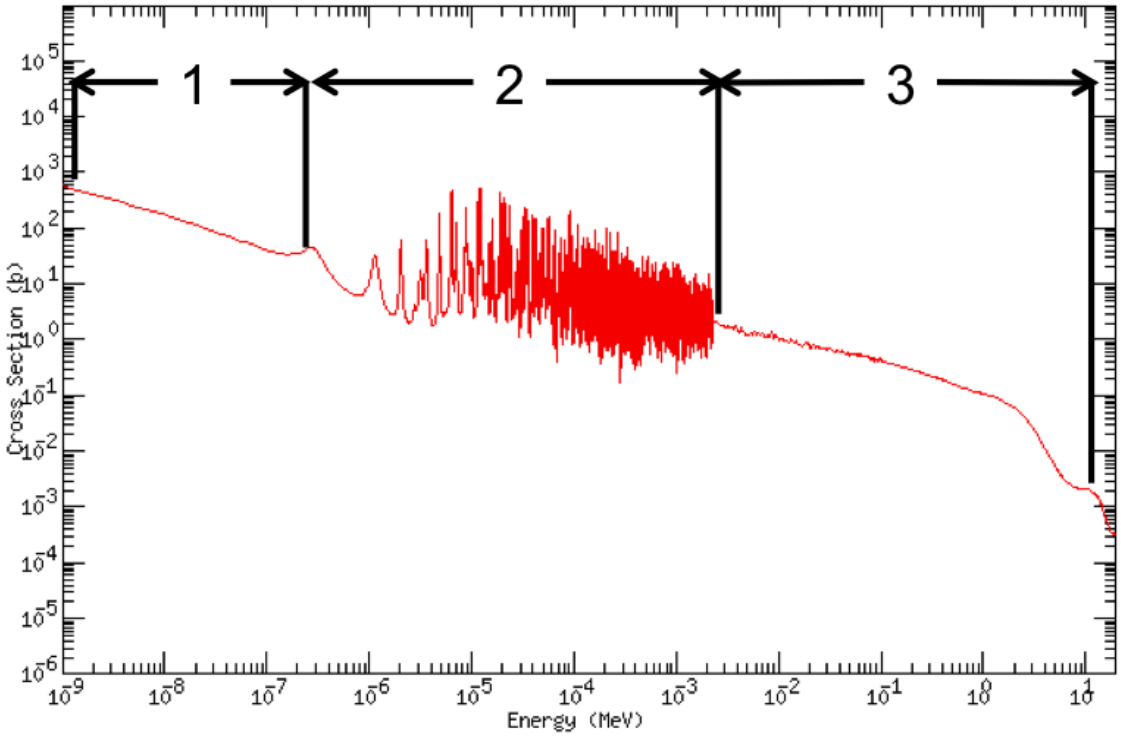


Figure 2.1 Absorption microscopic cross-section of ^{235}U [Chang, J.: ENDFPLOT 0.2 - Cross-section Plotter, KAERI. Available at: http://atom.kaeri.re.kr/endfplot.shtml.]

resonance region. Above the resonance region, the cross-section drops rapidly and smoothly to very small values. This region 3 is called the smooth region. Region 2 is usually present in heavy nuclei, because they are likely to absorb a neutron forming a compound nucleus in an excited state. The excited nuclei usually de-excites via a cascade emission of gamma rays, before it returns to the ground state [11].

2.2 Neutron Detection

Neutrons are generally detected through nuclear reactions that result in prompt energetic charged particles such as protons and alpha particles, because neutrons themselves are not charged particles and cannot ionize the detection medium.

2.2.1 Slow Neutron Detection

As mentioned above, neutrons are detected after causing nuclear reactions in the detection medium. It is important to point out the possible reactions that can be used to detect neutrons. Possible reactions between incident neutrons and the detection medium are shown in Figure 2.2.

All the reactions are sufficiently exothermal so that the kinetic energy of the reaction products is determined solely by the Q value of the reaction and does not reflect the incoming energy of the slow neutron [11]. In most of the fast neutron detection tasks, the measurement of incident neutron energy is required, which is not a goal in slow neutron detection. The measurement of incident neutron energy is often called neutron spectrometry.

Many kinds of slow neutron detectors have been invented and used widely for years. Here are some examples of the most popular detectors: the BF_3 tubes based on the boron-neutron reaction; the ^3He proportional counters based on the ^3He (n, p) reaction; and fission counters based on Uranium fission reaction.

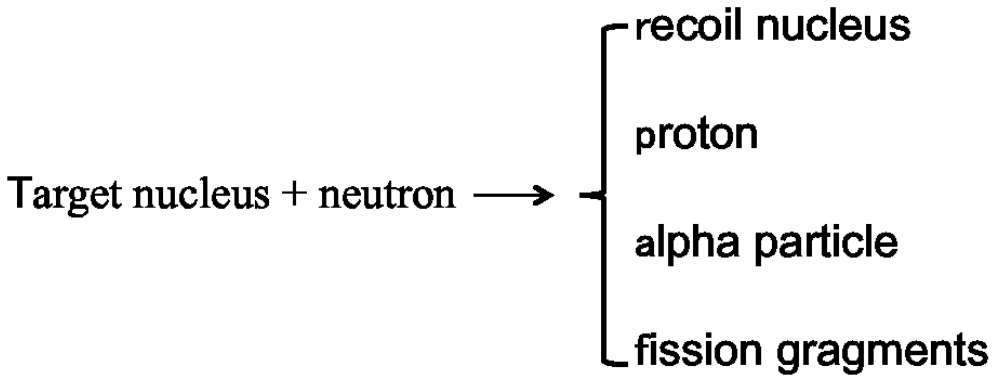


Figure 2.2 Possible reaction products used to detect neutron

2.2.2 Fast Neutron Detection

Basically, the mechanism of the low energy neutron detection can all be applied to fast neutron detection. However, as discussed in Section 2.1.2, the probability for fast neutrons to be absorbed by the detection material and undergo the reactions shown in Fig. 2.2 is quite small. This leads to very low detection efficiency in these detectors. As a result, additional mechanisms should be exploited to detect fast neutrons.

The Lithium glass scintillator and the ^{3}He scintillator are important examples of neutron spectrometers based on fast neutron-induced reactions. The limitations discussed above can be overcome when the fast neutron is made to induce direct suitable nuclear reactions. Helium-3 is mainly from the decay of Tritium, which is rare in natural abundance. Governments produce tritium for use in nuclear weapons. Thus, the amount of helium production is determined by the needs of nuclear weapon, not the needs of Helium-3 itself. Although Helium-3 is excellent for neutron detection due to the high sensitivity, increasing demand of Helium-3 of governments and other research organizations has made it rare and expensive.

Another important issue associated with fast neutron detection is elastic neutron scattering. An incident fast neutron transfers fraction of its energy to the scattered nucleus by elastic scattering. After its energy decreases into a lower energy range, it has a higher probability to be absorbed. It then can be detected through the methods used to detect slow neutrons.

For incoming neutrons with nonrelativistic kinetic energy, the energy of the recoil nucleus E_R (laboratory system) can be calculated using the following equation:

$$E_R = \frac{4A}{(1+A)^2}(\cos^2\theta)E_n \qquad \text{Eq. 2.3}$$

where A is the mass of target nucleus mass, E_n is the incident neutron energy (laboratory system), and θ is the scattering angle of the recoil nucleus in laboratory system ($\theta \in [0, \pi/2)$). The scattering process is shown in Figure 2.3. The maximum recoil energy is

$$E_{R,max} = \frac{4A}{(1+A)^2}E_n \qquad \text{Eq. 2.4}$$

The maximum fractional energy transfer decreases as the target nucleus mass increases, which means that light nuclei, especially Hydrogen, play the predominant role in recoil detectors. Thus, neutrons can transfer most of its energy to hydrogen atoms in a single scatter on average (see comparison in Table 2.2).

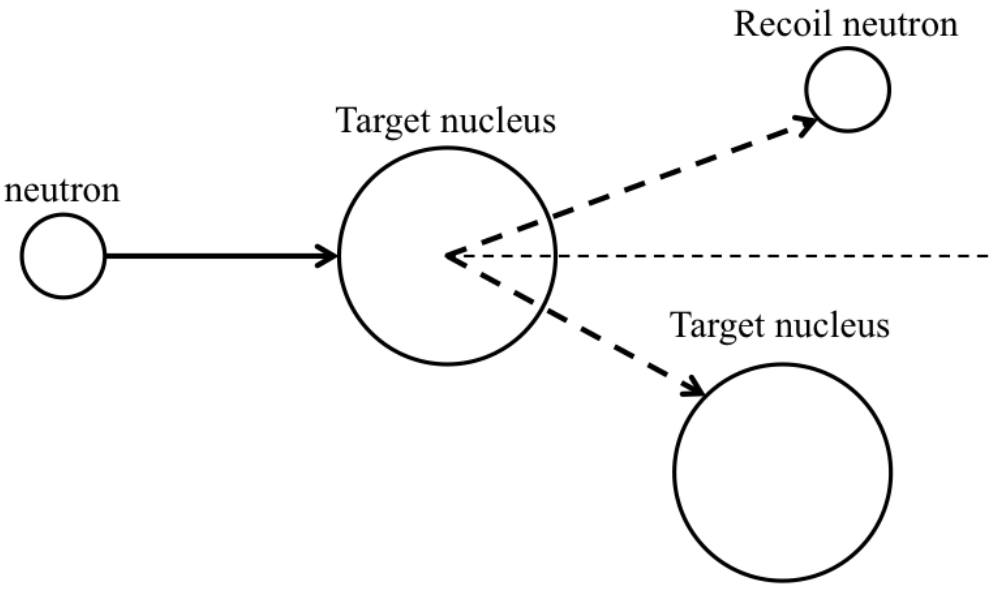


Figure 2.3 Neutron elastic scattering diagram (laboratory coordinate system).

Table 2.2 maximum fractional energy transfer in a single scatter

Target nuclei	A	$E_{R,max}=4A/(1+A)^2 E_n$
^{1}H	1	1
^{12}C	12	0.284
^{16}O	16	0.221

To measure recoil protons is one of the most straightforward ways to detect fast neutrons using scintillators containing hydrogen. Because the range of recoil protons is quite small in MeV range compared to the size of scintillator, they can deposit most of their energy inside the scintillator. This kind of scintillation detector has been studied and used before [12], [13].

2.3 Neutron Spectrometer

Neutron Spectrometer is an instrument used to determine the energies of neutrons and the relative intensities of neutrons of different energies in a neutron beam. Not all the neutron detectors can be used to measure the neutron energy spectrum. For example, the ^{10}B lined proportional counters are only used for the flux monitoring rather than measure the energy spectrum.

Neutron spectrometers are widely used in many fields. The main application areas are listed and briefly described below:

- Fusion: a neutron spectrometer provides the fusion research area a valuable diagnostic tool. The detailed spectrum near 14.1 MeV gives the information about the ion temperature and the degree to which reaction products have been incorporated in the plasma [14] [15].

- Laboratory usages: a neutron spectrometer is important to obtain the neutron energy spectrum when neutron instruments are calibrated, where neutron cross-sections are measured, and where nuclear physics experiments are performed.
- Special nuclear materials: the application area of special nuclear materials includes problems such as arms control, transuranic waste characterization, material safeguards, and facility decontamination and decommissioning. These applications collectively require instruments that can sensitively detect or characterize uranium and plutonium isotopes under conditions that can be challenging for a variety of different reasons [16].

Neutron spectrometry is difficult for 2 main reasons: a neutron cannot ionize other particles, so it has to transfer all of its energy to ionizing particles and the energies of the ionizing particles must be measured; it is hard to find a mechanism that is feasible for neutron spectrum measurement of the whole neutron energy range. Conventional neutron spectrometers are generally based on the techniques listed in Table 2.3.

2.4 Organic Scintillation Detectors Mechanism

Scintillation detectors are widely used to obtain the energy spectrum of incident radiation. A scintillation detector usually consists of a scintillator, a photomultiplier tube (PMT), and associated electronic equipment. The general configuration of a scintillation detector is shown in Figure 2.4. A scintillator is a material that exhibits scintillation – the property of luminescence when excited by ionizing radiation. It has many desired properties, such as high density, fast operation speed, and low cost. The photomultiplier

Table 2.3 Conventional neutron spectrometry techniques

Techniques	Examples
Time-of-flight	IN5
Nuclear reaction	^{3}He(n, p)T, ^{6}Li(n, α)T
Nuclear recoil	Proton recoil scintillators

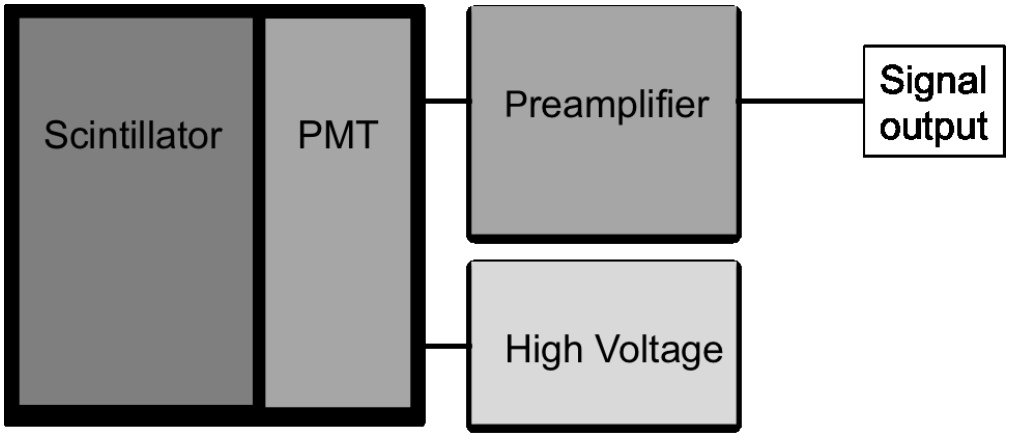


Figure 2.4 The general configuration of a scintillation detector.

tube absorbs the optical photons emitted by the scintillator and emits electrons via the photoelectric effect. These electrons are then multiplied by the following components, resulting in an electrical pulse, which carries important energy information about the incident particle.

According to the composition, scintillators are classified into organic scintillators and inorganic scintillators. Organic scintillators are more advantageous than inorganic scintillators because organic scintillators have much shorter decay time and lower density. Organic scintillators are generally useful for direct detection of electrons and alpha particles and also adaptable for gamma rays and neutrons [17]. The scintillation mechanism in organics is described below.

Molecular transitions are responsible for the production of scintillation light in organic scintillators. The ground state of a molecule is at point A_0, which coincides with the minimum of the potential energy. Ionizing radiation passing through the scintillator may give energy to the molecule and raise it to an excited state, i.e., the transition from A_0 to A_1, may occur. A_1 is not the stage of minimum energy. The molecule will release energy through lattice vibrations (that energy is eventually dissipated as heat) and move to point B_1. The point B_1 is still an excited state and, in some cases, the molecule will undergo the transition from B_1 to B_0, accompanied by the emission of a photon with energy equal to E_{B1}- E_{B0}. This transition, if allowed, takes place at times on the order of 10^{-8} s. It should be noted that the energy of the emitted photon (E_{B1}- E_{B0}) is less than the energy that caused the excitation (E_{A1}- E_{A0}). This difference is very important because otherwise, the emission spectrum of the scintillator would completely coincide with its absorption spectrum and no scintillation light would be produced [18].

Time response is a basic property of scintillators. If τ represents the decay time of prompt fluorescence, then the prompt fluorescence intensity at time t after excitation is:

$$I = I_0 e^{-t/\tau} \qquad \text{Eq. 2.5}$$

One approach assumes that the population of optical levels is also exponential [19] [20], so the overall shape of light pulse is given by

$$I = I_0 (e^{-t/\tau} - e^{-t/\tau_1}) \qquad \text{Eq. 2.6}$$

where τ_1 is the time constant describing the population of the optical levels. One of the most important differences between organic and inorganic scintillators is the decay time τ. In organic scintillators, τ is usually a few nanoseconds, while in inorganic scintillators, the decay time varies from 10^2 nanoseconds to 1 microsecond, which is much longer than in organic scintillators.

Light output is another important parameter of a scintillator. In a scintillator, only a small fraction of the energy loss of a charged particle in the scintillator is converted into visible light. This fraction (scintillation efficiency) of particle energy depends on both particle type and its energy. In most organic scintillators, the response to electrons is linear for particle energies above about 125keV [21].. This relation can be described by the following equation:

$$\frac{dL}{dx} = S\,\frac{dE}{dx} \qquad \text{Eq. 2.7}$$

where S is the scintillation efficiency, dL/dx is the fluorescent energy emitted per unit path length, and dE/dx is the specific energy loss of the charged particle. However, Birks assumes that some fraction of the fluorescence will lead to quenching. To account for the probability of quenching, Birks then writes:

$$\frac{dL}{dx} = L_0\,\frac{\frac{dE}{dx}}{1+k_B\frac{dE}{dx}} \qquad \text{Eq. 2.8}$$

Eq. 2.8 is referred to as Birks' law [22] [23]. The k_B is called the Birks' constant.

Capability of pulse shape discrimination is also a crucial feature of scintillators. Although the majority of optical photons are produced shortly after the excitation, a slow component can be observed sometimes, which usually has a decay time of several hundred nanoseconds. It would be significant when used for pulse shape discrimination, because the fraction of light produced as the slow component depends on the nature of the incident particle. Pulse shape discrimination is widely used because it is necessary to discriminate against events produced by gamma rays when organic scintillators are utilized to detect neutrons. Details about the pulse shape discrimination methods will be discussed in Section 2.6.

2.5 Capture-Gated Neutron Spectrometry

The neutron spectrum produced by a mono-energetic neutron beam in a typical organic scintillator based on neutron scattering shows a broad continuum that stretches from zero up to the equivalent of the full neutron energy, as shown in Figure 2.5 [24].

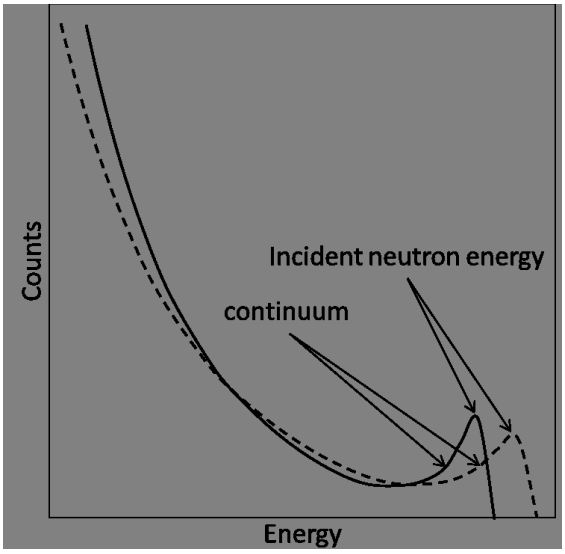


Figure 2.5 Example of neutron spectrum for a recoil proportional counter

This continuum shows the fact that most of the incident fast neutrons escape the scintillator before depositing their full energy. On the other hand, the ideal neutron spectrum should be a single full energy peak for mono-energetic neutrons. If a method is implemented to discard the events in which the neutron escapes without losing all its kinetic energy, a full energy single peak could be expected. This becomes the motivation that fostered the researchers to develop the capture-gated method.

Scintillators operated in capture-gated mode are normally loaded with a neutron-absorbing material, such as ^{10}B, ^{6}Li, and ^{113}Cd. The principle of the capture-gated method is illustrated in Figure 2.6. The example is given using a ^{10}B-loaded scintillator.

In a ^{10}B-loaded scintillator, due to the small absorption cross-section (see Figure 2.7a) at high energy range and the large elastic scattering cross-section (see Figure 2.7b), the incident fast neutron first undergoes several collisions, producing recoil protons instead of being directly absorbed by Boron-10. After the neutron undergoes multiple collisions and is slowed down into the thermal range, it continues to diffuse as a thermal neutron within the scintillator. It will then be captured by ^{10}B instead of escaping as in normal scintillators if sufficient boron concentration is present.

Figure 2.6 Principle of capture-gated method using a scintillator loaded with ^{10}B.

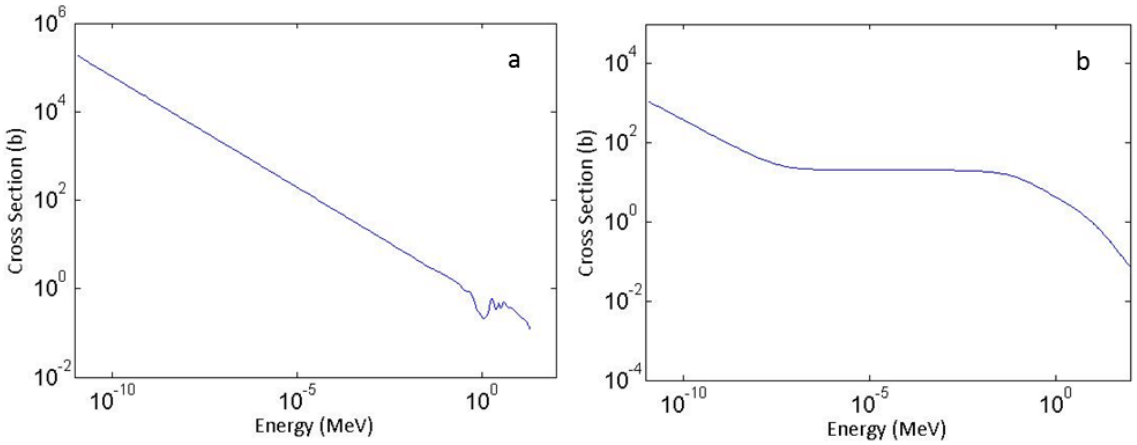


Figure 2.7 Total absorption cross-section of boron-10 (a) and elastic scattering cross-section of Hydrogen (b) [Chang, J.: ENDFPLOT 0.2 - Cross-section Plotter, KAERI. Available at: http://atom.kaeri.re.kr/endfplot.shtml.].

Because the fast neutron travels in the scintillator at a high speed, all the recoil protons are produced within a short period of time (typically less than 50 ns). Therefore, the pulse height, related to the sum of the recoil protons energies, can be used to estimate the full energy of the incident neutron. After captured, neutron interacts with boron-10 through the following reaction:

$$^{10}B + n \rightarrow \alpha + {}^{7}Li + \gamma\,(480keV) \qquad \text{Eq. 2.9}$$

with Q = 2.310 MeV (94%) or

$$^{10}B + n \rightarrow \alpha + {}^{7}Li \qquad \text{Eq. 2.10}$$

with Q = 2.792 MeV (6%). The Q value of the reactions equals to the total kinetic energy of the generated alpha particle and Lithium ion, which then deposit their energy locally

and give rise to a second pulse – the neutron-capture pulse. The time between the neutron-capture pulse and the recoil-proton pulse is called the capture time, typically around 10 microseconds. Therefore, once the capture pulse is detected, the neutron can be considered to have lost most of its energy to recoil protons, as only neutrons in the thermal energy range can be easily captured. Thus, the capture pulse becomes an important and unique symbol that can be used to screen out the events in which neutrons lose all of their energy in the scintillator.

The capture-gated method is described as follow. The expected sequence of pulses is shown in Figure 2.8. Once the capture pulse is detected, a time gate is opened. Only within this time window, the recoil-proton pulse is accepted as a 'true' signal. The accepted pulses are then used to reconstruct the neutron energy spectrum.

Neutron spectrometers operated in capture-gated mode can be constructed as a boron-loaded plastic scintillator [25], boron-loaded liquid scintillator [26], or ^{6}Li-loaded glass scintillator [27].

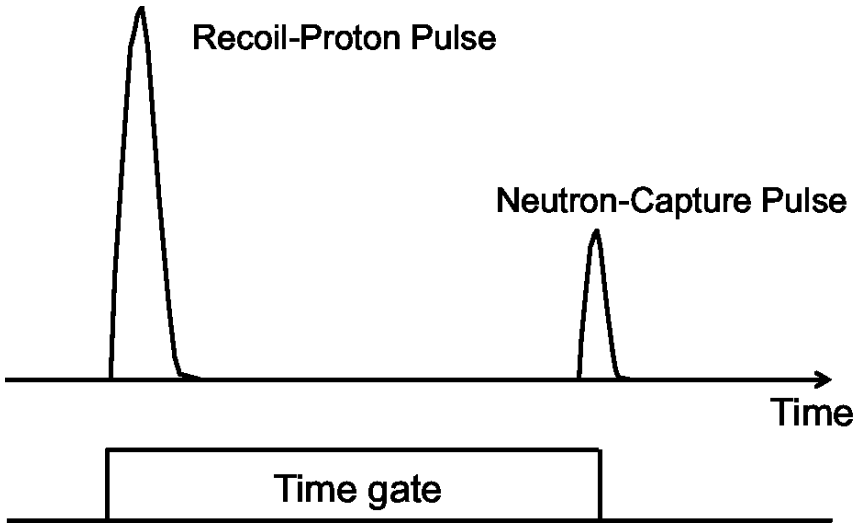


Figure 2.8 Discipline of capture-gated method.

2.6 Pulse Shape Discrimination Algorithms

The slow component fraction should depend primarily on the specific ionization density of the charged particle (commonly denoted LET or dE/dx). So the relative intensities of the fast and slow components is also decided by LET. This constitutes the basis for Pulse Shape Discrimination (PSD) because of the difference in the slow component induced by different types of radiation [23].

Liquid scintillators are a good choice for neutron spectrometry because of their large neutron stopping power and good pulse shape discrimination capabilities. Because of the difference in their linear energy transfer values, gamma rays (via secondary electrons) and neutrons (via heavy charged particles) generate scintillation pulses with a very different pulse shape. Currently, pulse shape discrimination in a liquid scintillator uses one of the following three analog algorithms [28]:

1) rise-time inspection, which determines the time at which the light pulse reaches a certain fraction of its maximum amplitude;
2) the zero-crossing method, which allows an indirect evaluation of the rise time of the pulse performed by measurement of time for a bipolar signal to cross the baseline;
3) charge comparison, which measures the integrated portions of the signal in the fast and slow components of the light pulse.

The 10–90% pulse rise-time algorithm and the pulse ‘time over threshold’ algorithm are the two commonly used rise-time algorithms. They are shown in Figure 2.9 [29]. They both belong to the rise-time inspection method. Figure 2.9 (a) is an integrated pulse

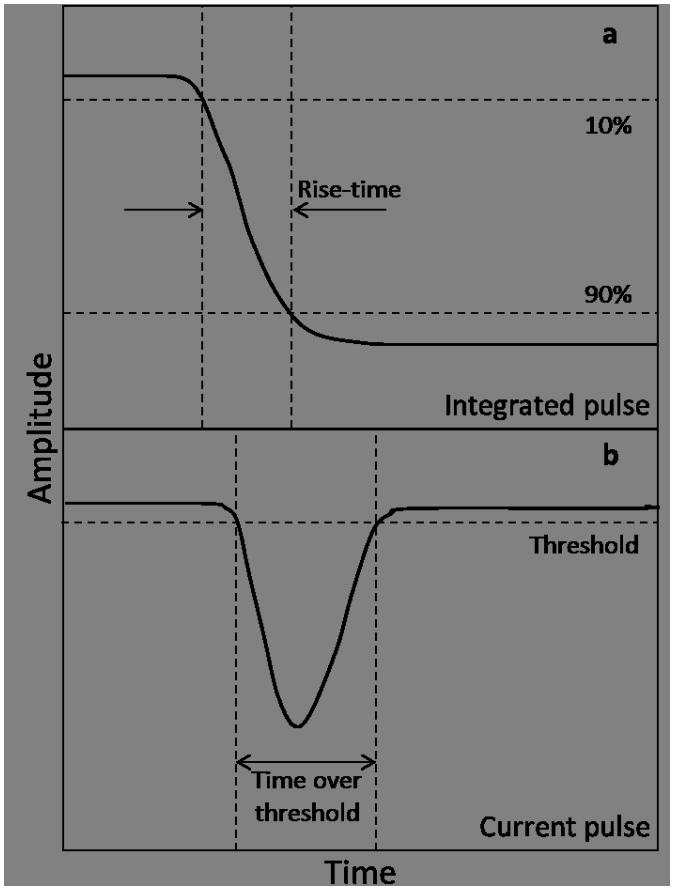


Figure 2.9 Pulse rise-time and pulse time over threshold PSD algorithms.

from the preamplifier, and Figure 2.9 (b) is a current pulse directly from the photomultiplier tube. In each case, the pulse rise-time and pulse time over threshold algorithm is applied, respectively.

The charge comparison algorithm is also widely used. In this algorithm, each pulse is integrated via two separate routes. The first integration, called the total integral, is from the beginning to an optimized end point of the tail. The second integral, taken from a certain starting position on the tail after the pulse's maximum to the same end point as used for the total integral, is called the tail integral. The ratio of the rail integral to the total integral is used to discriminate events resulted from different particles [30] (see Figure 2.10).

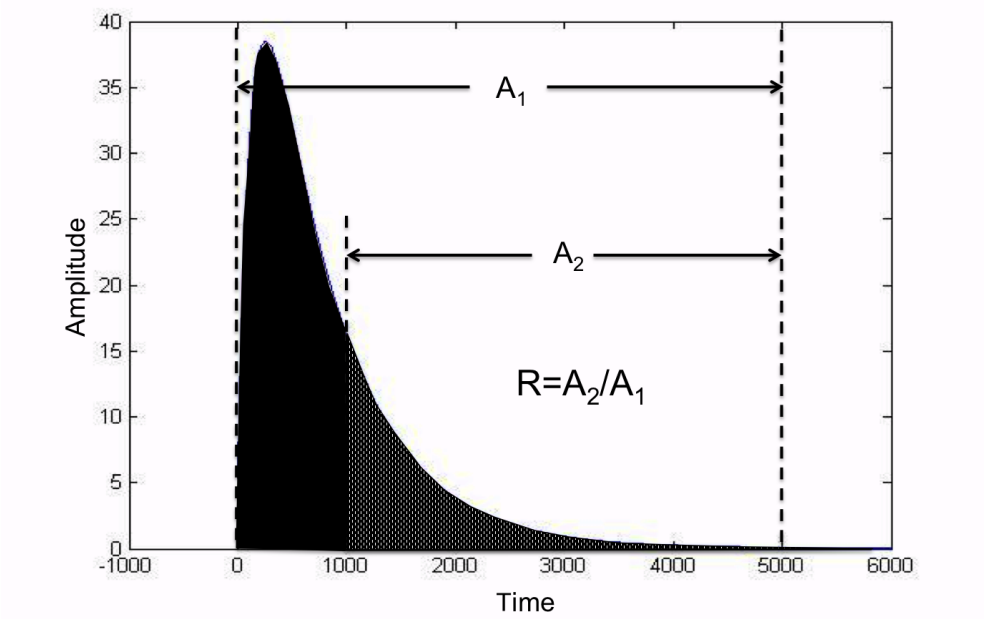


Figure 2.10 Description of a charge integration algorithm.

2.7 Purposes of the Thesis

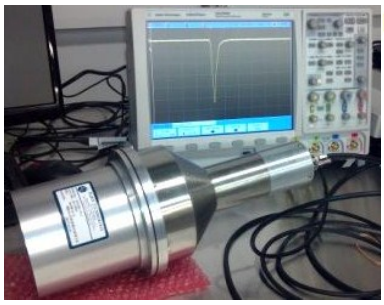
In neutron spectrometry with capture-gated technique, the prompt recoil proton pulse from an incident fast neutron is gated by the boron capture pulse produced by the same neutron after being thermalized within the scintillator. It is always assumed that neutron absorption only happens after the incident neutron loses all of its kinetic energy so that the recoil-proton pulse amplitude can be used to derive the incident neutron energy. However, this assumption is not always correct. As pointed out in previous research, the neutron energy at capture is not constant: it is a function of incident neutron energy [31]. The average energy at capture increases with incident neutron energy. Therefore, a discrepancy exists between the energy obtained based on the recoil-proton pulse and the actual incident neutron energy. This discrepancy can cause the degradation of energy resolution in a neutron spectrum. In this thesis, we will demonstrate the degradation of energy resolution and propose a method to correct it.

Furthermore, we will examine the Pulse Shape Discrimination feature of the loaded scintillator, since the correction is strongly based on accurate discrimination between neutron-capture pulses, recoil-proton pulses, and gamma pulses.

2.8 Method

To implement the correction, we assume the energy discrepancy is correlated with the capture time - elapsed time between the neutron scattering pulse and the subsequent neutron capture.

The detector used in this study is a 5" by 5" EJ-339A boron-loaded liquid scintillator manufactured by Eljen Technology, as shown in Figure 2.11. Its composition is shown in Table 2.4.



PROPERTIES

^{10}B Content, weight percent	See Table Below
Density, g/cc (20^{O}C)	0.92
Light Output, (% anthracene)	65%
No. of Blue Photons per 1 MeV electron	10,000
Wavelength of Maximum Emission	425nm
Decay Time, short component	3.7 ns
Refractive Index	1.415
Flash Point	16^{O}F (-8^{O}C)

Figure 2.11 The EJ-339A detector used in this study and its properties.

Table 2.4 Atomic Composition of EJ-339A (atoms/cc x 10^{22})

Boron-10	0.053
Carbon	2.9
Hydrogen	5.03
Oxygen	0.814

To study the relation between the capture time and the energy discrepancy discussed in Section 2.7, we used Geant4 to build an EJ-339A model. In the simulation, the detector is bombarded with a fast neutron beam of a certain energy. Two variables were generated via the Geant4 simulation: the energy of the neutron (E) right before being absorbed and the elapsed time (Δt) between two pulses. The relation between these two variables was then studied using a customized algorithm in MATLAB. Simulations of optical process in this scintillator were also executed in order to inspect the Pulse Shape Discrimination properties of EJ339A.

Geant4 is a toolkit for simulating the passage of particles through matter. It includes a complete range of functionality including tracking, geometry, physics models, and hits. The physics processes offered cover a comprehensive range, including electromagnetic, hadronic and optical processes, a large set of long-lived particles, materials, and elements, over a wide energy range starting, in some cases, from 250 eV and extending in others to the TeV energy range. It has been created exploiting software engineering and object-oriented technology and implemented in the C++ programming language. It has been used in applications in particle physics, nuclear physics, accelerator design, space engineering, and medical physics [32].

After the simulation, we conducted some experiments to compare the results with simulation results. In the experiments, a digitizer was used to obtain the pulse shape data and MATLAB was then used to analyze the data.

CHAPTER 3

SIMULATIONS AND EXPERIMENTS

3.1 Spectrum Analyses and Correction

The aim of the work presented in this chapter was to investigate the assumption that a discrepancy exists between the energy obtained based on the recoil-proton pulse and the actual incident neutron energy and it causes the degradation of energy spectrum resolution, and the relation between the energy discrepancy and the time interval between the first scattering and the final capture event. Furthermore, a spectrum correction algorithm based on the relation was developed.

In this section, the Geant4 Monte Carlo simulation toolkit was used as the main method to accomplish the work. All the simulations were based on the geometry and material property of the EJ339A Boron-loaded liquid scintillator

3.1.1 Simulation Setup

The boron-loaded liquid scintillator, EJ339A, is a 5" by 5" cylinder. The boron present in the EJ339A is enriched with ^{10}B to 90%, with the total boron in the glass representing 5.89 % of the weight. Since the neutron scattering and absorption reactions both happen inside the scintillator, and the light collection by PMT is not considered in this part of study, the PMT is not simulated in this part.

In this Geant4 simulation, the geometry was set up as the following:

- The z-axis is along the axis of the scintillator and the PMT (Photomultiplier tube);
- A neutron beam is directed to bombard the detector parallel to z-axis.
- The scintillation material is defined according to the composition given in Table 2.2.

The configuration is shown in Figure 3.1, and Table 3.1 describes the geometry specifications of EJ339A scintillator.

The DetectorConstruction class was used to specify the geometry and the material components of EJ339A. The code for building the scintillator is shown in Figure 3.2.

The basic reaction processes associated in this simulation are the neutron scattering and absorption. The built-in Geant4 physics list QGSP_BERT_HP [32] was utilized, which uses Geant4 Bertini cascade for the primary protons, neutrons, pions, and Kaons below 10GeV and the data-driven high-precision neutron package (NeutronHP) to transport neutrons below 20 MeV down to thermal energies.

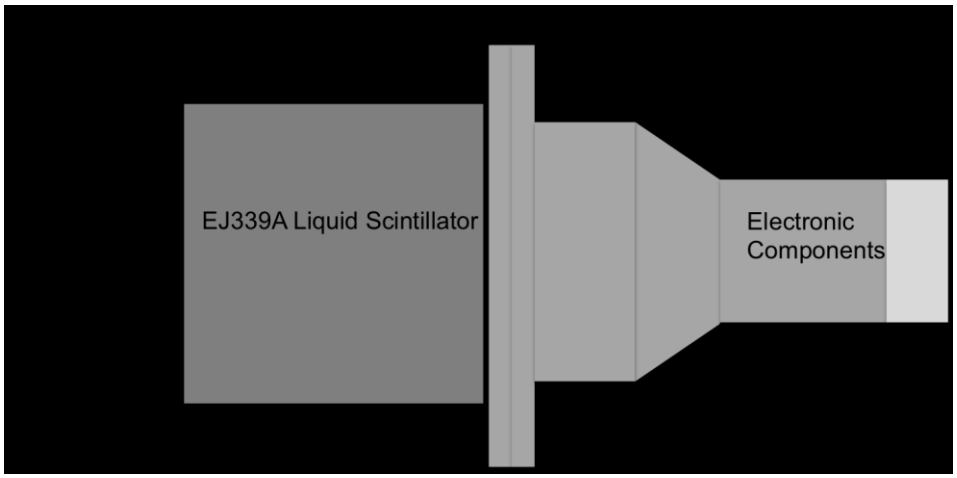


Figure 3.1 Geant4 simulation setup: (a) the detector geometry; (b) radiation transport.

Table 3.1 Geometry Specifications Of EJ339A Scintillator.

Scintillator	EJ339A
Shape	Cylinder
Diameter	12.7 cm
Height	12.7 cm
Density (g/cm^3)	0.92
Components	^{10}B, ^{11}B, C, H, O

```
G4Tubs*            solidScint;   // pointer to the solid Scintillator
G4LogicalVolume*   logicScint;   // pointer to the logical Scintillator
G4VPhysicalVolume* physiScint;   // pointer to the physical Scintillator

G4ThreeVector positionScint =
    G4ThreeVector(0, 0, fScintThickness/2.0+fShellThickness+fAirGap);

solidScint = new G4Tubs("Liquid", 0.*cm, fScintDiameter/2.0,
     fScintThickness/2.0, 0.*deg, 360.*deg);
logicScint = new G4LogicalVolume(solidScint,ScintMater,"Liquid",0,0,0);
physiScint = new G4PVPlacement(0,               // no rotation
                               positionScint,   // at (x,y,z)
                               logicScint,      // its logical volume
                               "Liquid",         // its name
                               logicWorld,       // its mother  volume
                               false,            // no boolean operations
                               0);               // copy number
```

Figure 3.2 The code for building the scintillator geometry

In the PrimaryGeneratorAction class, a particleGun was defined as the neutron source that generates a mono-energy neutron beam hitting perpendicularly to the bottom surface of the scintillator. In each event, only one neutron particle was generated so that we can analyze the process in each single neutron reaction. The source is placed five centimeters away from the entrance window of the scintillator on the z-axis.

In the SteppingAction class, the energy, and important parameters of recoil protons and neutrons before capture are respectively obtained and written into two files –

neutron.gt4 and proton.gt4 – in each event (each incident neutron). The energy of pre-step and energy of current point are obtained by GetKineticEnergy function in each step. ParentID, TrackID, and EventID are obtained via GetParentID, GetTrackID, and GetEventID functions, respectively. By limiting the ParentID of proton to 1, which means the proton is generated by incident neutron, only the parameters of recoil protons are recorded. By setting the energy of the neutron to 0, which means the neutron is captured by boron and loses all its energy, only the parameters of the captured neutron are recorded. The EventID was used to associate the recoil protons and captured neutrons in the same event. Only the parameters of neutrons and protons inside the liquid scintillator were obtained and written, which means the escaping neutrons were not taken into account. Only in this case can the capture-gated mode work.

3.1.2 Simulation Results

A 0.1 MeV neutron beam was firstly generated to test the scintillator. Five million events (neutrons) were simulated. After the simulation, two files containing the energy and time of each concerned particle were analyzed. MATLAB was used to calculate the capture time (Δt) of each incident neutron by subtracting the generation time of the first recoil proton from the time when the neutron was captured. The concept of capture time is shown in Figure 3.3. Then, a 2-D histogram was created for the logarithm of capture time (Δt) and the logarithm of the capture energy ($E_{capture}$) – neutron energy before capture, which was easily read from the output file of neutron. The histogram is shown in Figure 3.4.

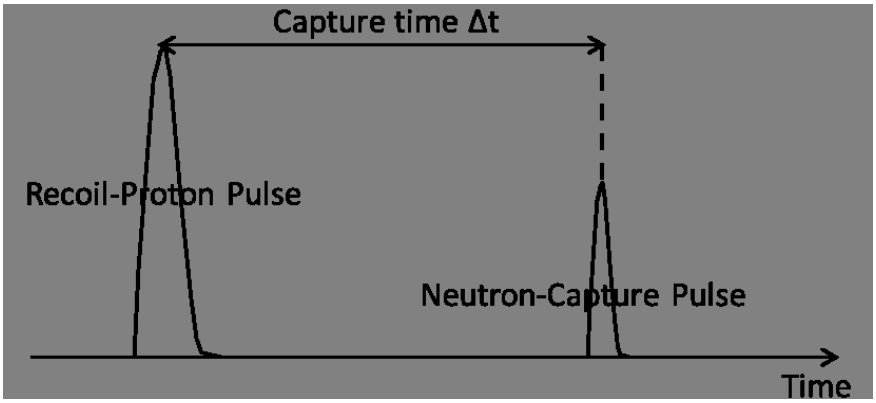


Figure 3.3 The concept of capture time

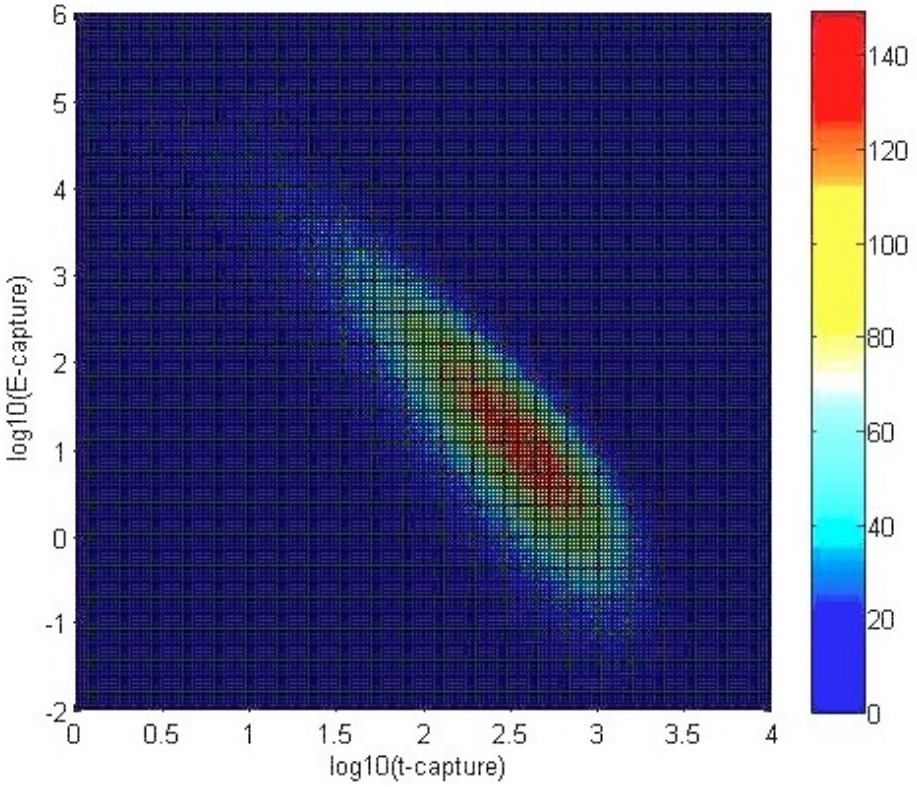


Figure 3.4. The relation between Δt (x-axis) and $E_{capture}$ (y-axis). The color is used to indicate the frequency for each $E_{capture}$-Δt pair.

From Figure 3.4, it can be observed that the energy discrepancy (i.e., $E_{capture}$) is correlated with the capture time (Δt). Generally, the capture energy decreases as the capture time increases, and the logarithm of the average energy under certain Δt is approximately inversely proportional to the logarithm of Δt. Moreover, for a particular value of Δt, $E_{capture}$ follows a certain distribution, as shown in Figure 3.5. However, the relation between the capture time (Δt) and the capture energy ($E_{capture}$) discussed above is only tenable when the incident neutron energy is 0.1 MeV. In order to correct the neutron energy based on the energy, we have to derive the $E_{capture}$-Δt relation for all incident neutron energy. The $E_{capture}$-Δt relations for 10 MeV and 1 MeV neutrons are shown in Figure 3.6, which are compared with the relation for 0.1 MeV neutron.

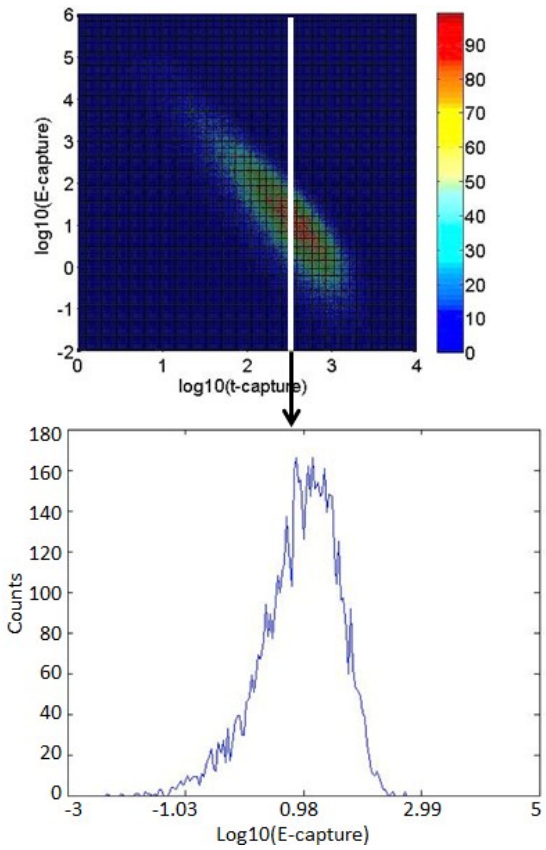


Figure 3.5 The capture energy distribution when $\log_{10}(\Delta t) = 2.5$

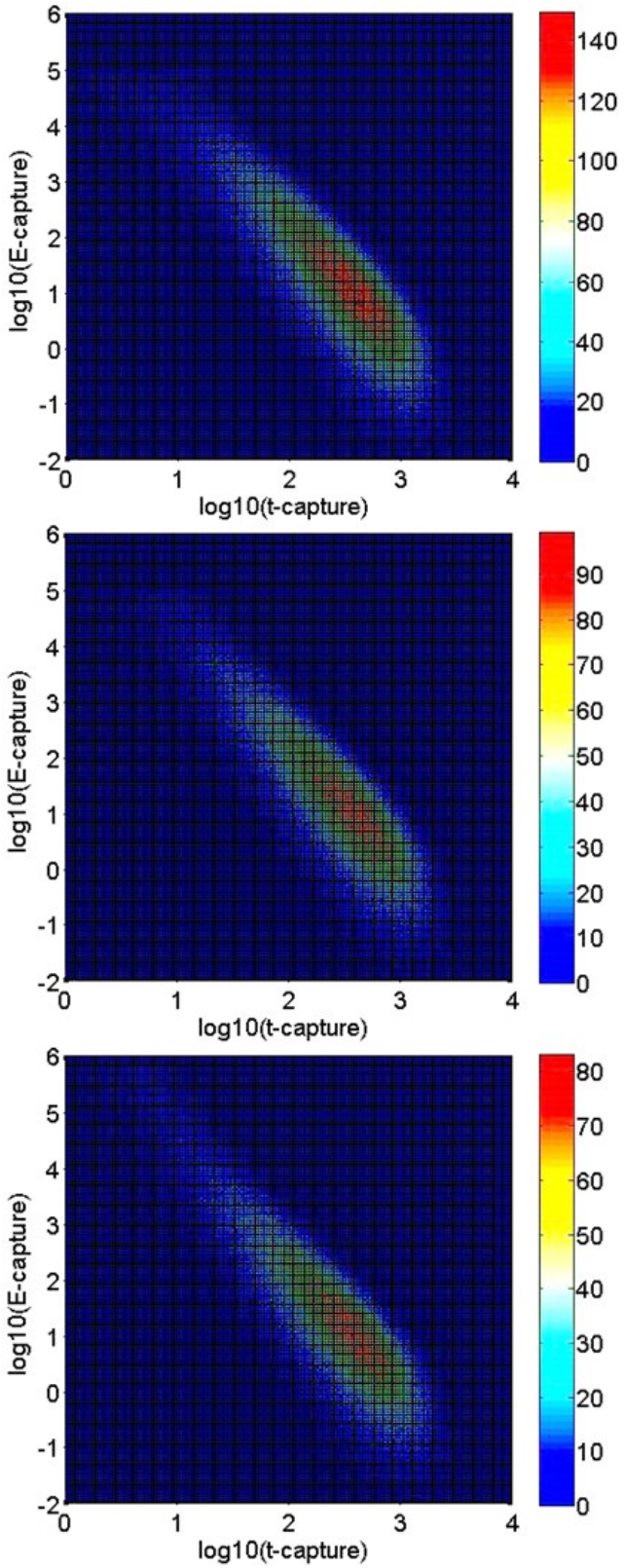


Figure 3.6 Comparison of $E_{capture}$-Δt relations for neutron energy 1 MeV (a), 10 MeV (b) and 0.1 MeV (the one at the top)

Figure 3.6 indicates that the relation between capture energy and capture time is consistent as the energy of the incident neutron changes. That means once the relation between Δt and $E_{capture}$ is obtained under a certain incident neutron energy, it can be used to correct the energy spectra for neutrons of all energy ranges.

3.1.3 Neutron Spectrum Correction

With the results of the simulation, the neutron energy spectrum can be corrected based on the capture time. As shown above, the capture time is correlated with the energy discrepancy. For a particular value of Δt, $E_{capture}$ follows a certain distribution centered at a value, as shown in Figure 3.5. Therefore, the distribution can be used to correct the energy discrepancy.

The algorithm is described as follow. First, the neutron energy spectrum was reconstructed by making the histogram of the incident neutron energy, which was the sum of recoil-proton energy. Second, for a $\log_{10}(\Delta t)$ value, the mean value of $\log_{10}(E_{capture})$ can be calculated through the relation between $\log_{10}(\Delta t)$ and $\log_{10}(E_{capture})$.

This relation is simply derived by fitting the relation of $\log_{10}(\Delta t)$ and $\log_{10}(E_{capture})$ to a linear function (Figure 3.7) using Excel. Once this relation is established, for each event, the mean capture energy corresponding to the value of $\log_{10}(\Delta t)$ can be identified. It is then added to the recoil-proton energy to get the corrected energy of the incident neutron. Then, the histogram of the corrected energies was made to form the corrected spectrum. The corrected and uncorrected energy spectrums for mono-energetic 0.1 MeV neutrons are shown in Figure 3.8. As shown in Figure 3.8, the uncorrected spectrum has a long tail in the lower energy range. This is caused by the fact that traditionally, the

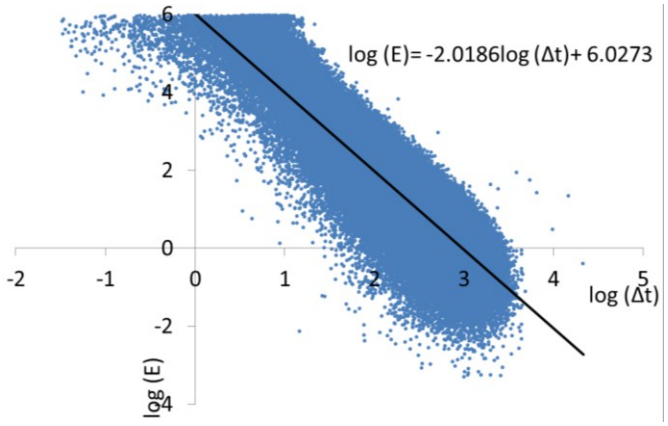


Figure 3.7 Fitted relation between $\log_{10}(\Delta t)$ and $\log_{10}(E_{capture})$

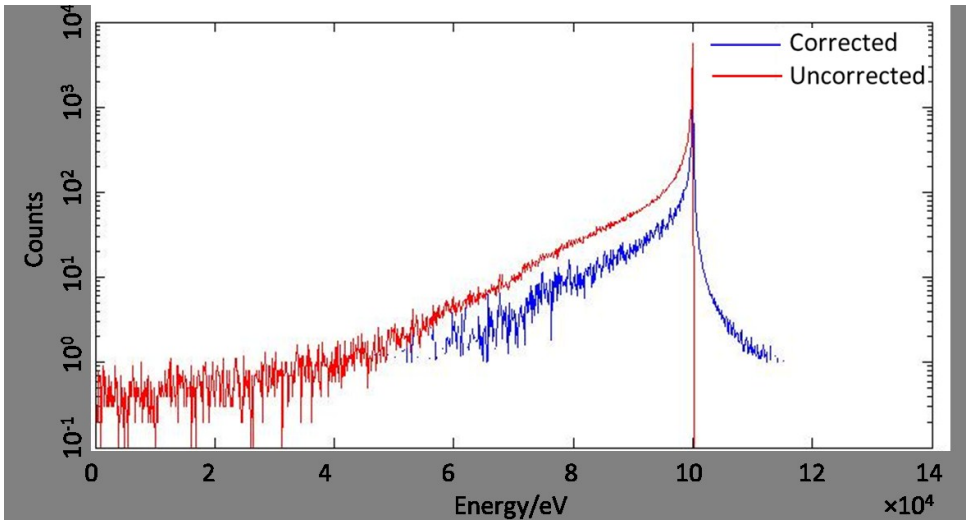


Figure 3.8 The corrected and uncorrected energy spectrums

thermal neutron energy right before capture is not taken into account. After being corrected, the tail is reduced. The correction helps to improve the energy resolution. However, the algorithm also introduces error. Since the $E_{capture}$ follows a certain distribution instead of being a single value, the algorithm that utilizes a mean value to substitute a distribution leads to inaccuracy. That is the reason for the small tail in the higher energy side of the full energy peak.

3.2 Pulse Shape Discrimination

Simulation using Geant4 is a good tool to study the Pulse Shape Discrimination performance of a detector. In order to obtain the pulse information after the incident radiation interacts with the scintillation material, the optical process of scintillation has to be integrated into simulation. In Geant4, there is no smooth transition between optical photons and gamma particle classes, so a separate physics process for optical photons has to be specified, which adds the scintillation process to the scintillator detector set-up.

3.2.1 Optical Process Simulation Setup

In the simulation of the optical process inside the scintillator, the same geometry configuration was used – the DetectorConstruction class. However, since the QGSP_BERT_HP Physicslist does not include the physics process for optical photons, the optical photon physics processes have to be added into the physics list. In Geant4, optical photons are different from gamma photons. A separate class named G4OpticalPhoton is to be used to model the transport of optical photons. The relevant optical processes in the simulation are: scintillation light generation, Cerenkov process, optical absorption, optical Rayleigh, and optical boundary processes.

In the DetectorConstruction class, the scintillation features were defined according the properties of the particular scintillator. As for EJ339A, the scintillation features are specified according to the emission spectrum shown in Figure 3.9 and the properties like the light output and number of photons generated per 1MeV of electron energy deposition, as shown in Figure 2.10. The code used to specify the scintillation feature is shown in Figure 3.10.

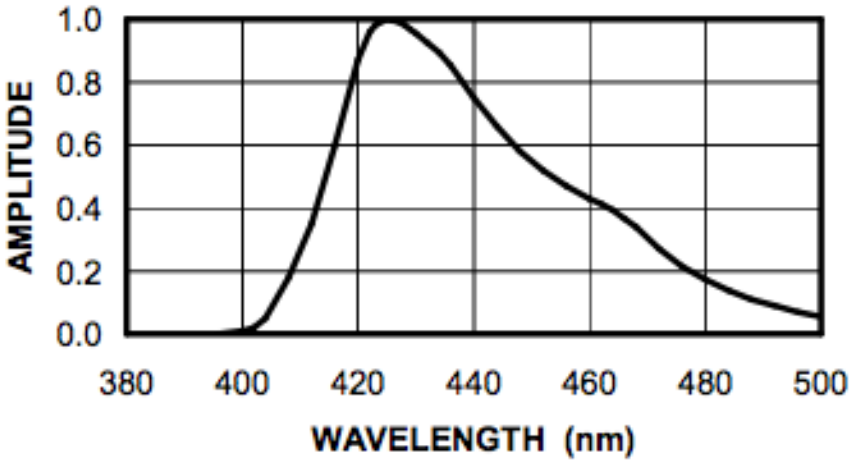


Figure 3.9 EJ339A emission spectrum

```
1. G4double naienergies[numentriesnai] =
2. { 2.48*eV,2.58*eV,2.69*eV,2.75*eV,2.82*eV, 2.85*eV, 2.88*eV, 2.92*eV, 2.94*eV,
   2.95*eV, 3.10*eV };
3. G4double naifastcomp[numentriesnai] =
4. { 0.05,  0.18,  0.44,  0.53,  0.75,  0.85,  0.95,  1.0,  0.93,
   0.88,  0.0 }; //maximum emmisson wavelenth: 425 nm (2.92eV)
5. naiprop->AddConstProperty("SCINTILLATIONYIELD",10.0/keV); //from the data sheet
6. naiprop->AddConstProperty("RESOLUTIONSCALE",1.0);
7. naiprop->AddConstProperty("FASTTIMECONSTANT",3.7*ns); // from the data sheet
8. naiprop->AddConstProperty("SLOWTIMECONSTANT",3.7*ns); // from the data sheet
9. naiprop->AddConstProperty("YIELDRATIO",0.65); //from the data sheet
```

Figure 3.10 Scintillation feature specification code of EJ339A in Geant4 DetectorConstruction class

Line 1-4 of the code in Figure 3.10 specified the emission spectrum of EJ339A shown in Figure 3.9. Line 5 defined the scintillation yield to 10 optical photons per keV, which can be found to be 10,000 blue photons per 1 MeV from Figure 2.10. The fast time constant, which means the short component decay time in Figure 2.10, was set to be 3.7 nanoseconds as given in line 7. Then in the 9th line, the yield ratio, or light output, was defined to be 0.65.

Furthermore, since the simulation includes the scintillation process caused by heavy particles in organic scintillators, the effect according to Birk's Law needs to be added. The organic scintillator does not respond linearly to the ionization density. The same amount of energy deposition does not always produce the same amount of light when different particles are considered. In Geant4, Birk's Law was implemented by adding G4EmSaturation into Physicslist class and setting Birk's constant in DetectorConstruction class, which is shown below.

In DetectorConstruction class, the code was added after the specification of scintillation feature:

"Scin->GetIonisation()->SetBirksConstant(0.126*mm/MeV);"

and in Physicslist class, the emSaturation instance must be defined as:

"G4EmSaturation*emSaturation=G4LossTableManager::Instance()-> EmSaturation()

theScintillationProcess->AddSaturation(emSaturation)."

3.2.2 Optical Process Simulation Results

To begin with, the scintillator is bombarded by a mono-energy gamma beam (480keV). The optical process simulation is visualized in Figure 3.11 based on the configuration described above.

In Section 2.4, the mechanism of the organic scintillation detector was discussed. Time response of an organic scintillator is given in Section 2.4 by:

$$I = I_0(e^{-t/\tau} - e^{-t/\tau_1}) \qquad \text{Eq. 3.1}$$

where τ_1 is the time constant describing the population of the optical levels. Thus, the

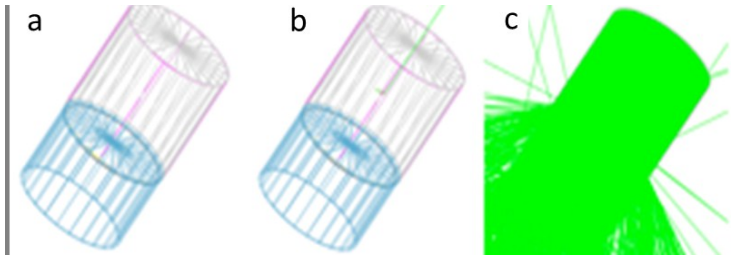


Figure 3.11 Optical photon transport in EJ339A

generation time of each photon is an important value in this simulation. The SteppingAction class was modified so that the time of every optical photon when generated can be obtained and recorded in the output file. The travel distance of the optical photon is shorter than the length of the scintillator, so the travel time of the photon before hitting on the cathode is about 0.25 ns. The generation time is about several nanoseconds, so the generation time of the optical photon can be used to estimate the time when the optical photon is collected by cathode. Then, the histogram of these times of optical photons was drawn using MATLAB. The histogram of photon time is close to the real pulse shape because the number of optical photons is proportional to the number of photoelectrons produced in the PMT, which is then proportional to the output pulse amplitude.

Afterwards, the histogram is fitted to the equation:

$$A = s_1 + s_2 \times e^{(-t/s_3)} + s_4 \times e^{(-t/s_5)} \qquad \text{Eq. 3.2}$$

using a MATLAB fitting function, and then the tail-to-total ratio is calculated as 100 ns, and the total range is from 0 to 100ns. The histogram and the fitted figure are shown in Figure 3.12.

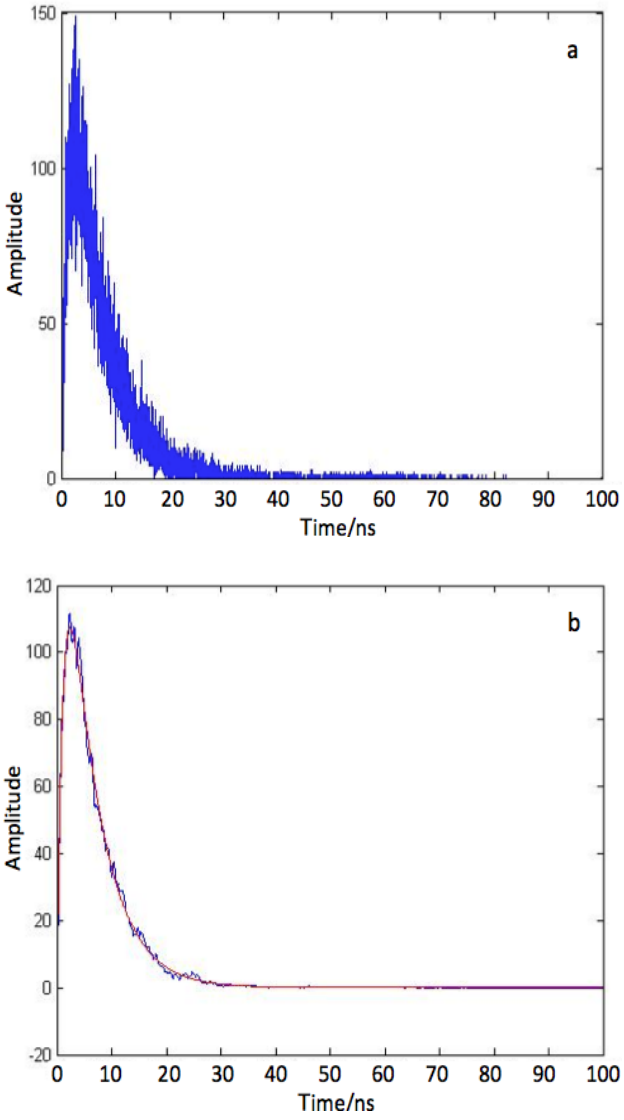
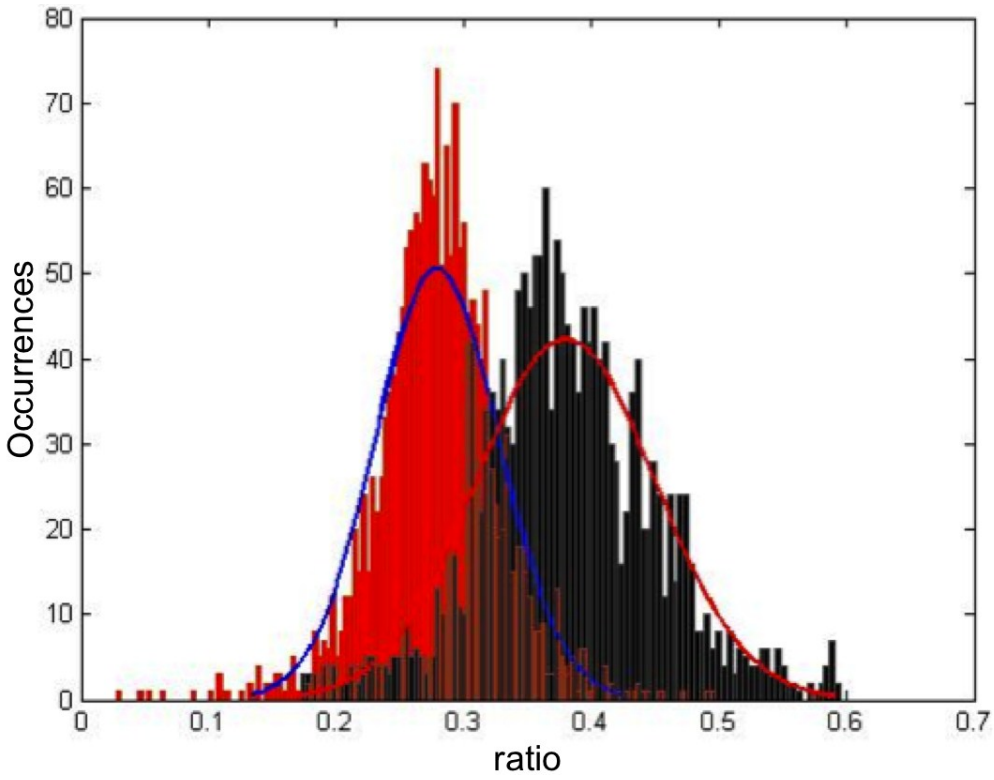


Figure 3.12 Histogram of photon times (a) and fitting results (b)

After the simulation of gamma ray, the scintillator was then bombarded by a fast neutron beam whose energy is 1 MeV. The tail-to-total ratio for pulses generated by recoil protons was calculated using the same method described above for gamma pulses. The tail-to-total ratio for gamma pulses was drawn and compared to ratio distribution for the recoil-proton pulses (see Figures 3.13). The histogram and fitting line on the left shows the ratio for gamma pulses; the ratio for the proton is shown on the right.



Figures 3.13 Comparison of tail-to-total ratio for gamma pulses and neutron pulses.

In Figure 3.13, we can see a separation of ratios for the recoil proton pulse and gammas pulse is observed as expected. It indicates that the Pulse Shape Discrimination feature is good. Although there is an overlapping area, it does not produce too much inaccuracy when doing pulse discrimination. So, based on the PSD feature derived via simulation, different pulses can be discriminated in EJ339A.

3.2.3 Experiment Results

An experiment to test the gamma pulse shape was conducted. The experiment configuration is shown in Figure 3.14. In the experiment, Cs-137 is used as the gamma source. The PMT was connected to 1060 V High Voltage and then the signals from the PMT were acquired by using the NI system with LabVIEW software.

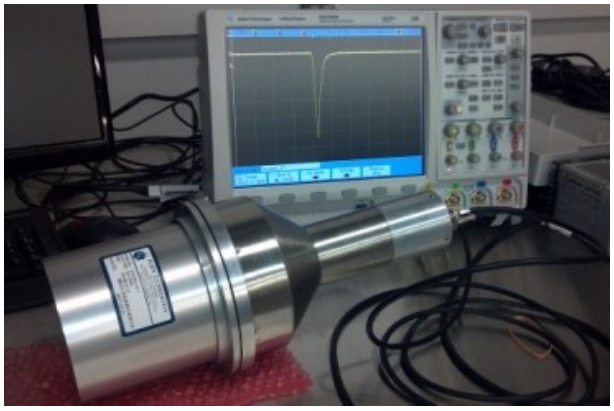


Figure 3.14 Experiment configuration

A 14-bit 100MS/s digitizer, NI PXIe-5122, was used to acquire gamma pulse data. The National Instruments 5122 high-speed digitizers feature two 100 MS/s simultaneously sampled input channels with 14-bit resolution, 100 MHz bandwidth, and up to 512 MB of memory per channel in a compact, 3U PXI Express, PXI, or PCI device.

The output file is a binary file containing the pulse shape information. The output file can be read by MATLAB and the gamma pulse shape can be reconstructed, which is shown in Figure 3.15.

After the gamma pulses data were read by MATLAB, the rise-time of these gamma pulses were easily calculated. Then, the histogram of the calculated rise-times was made to obtain the rise-time distribution, which is shown in Figure 3.16

From Figure 3.16, we can say the rise-times of measured gamma pulse were centered at 10.1 ns with a very small deviation. That is a good indication for a good n/γ pulse shape discrimination property because the range of gamma pulse rise-time is very narrow. After the experiment with the neutron source, the full pulse shape discrimination feature would be gained.

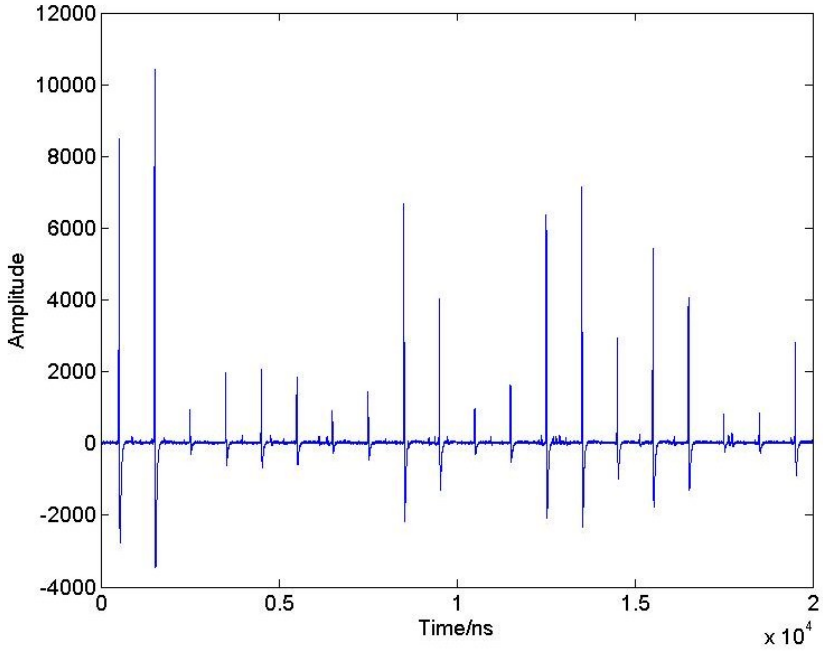


Figure 3.15 Reconstructed gamma pulses

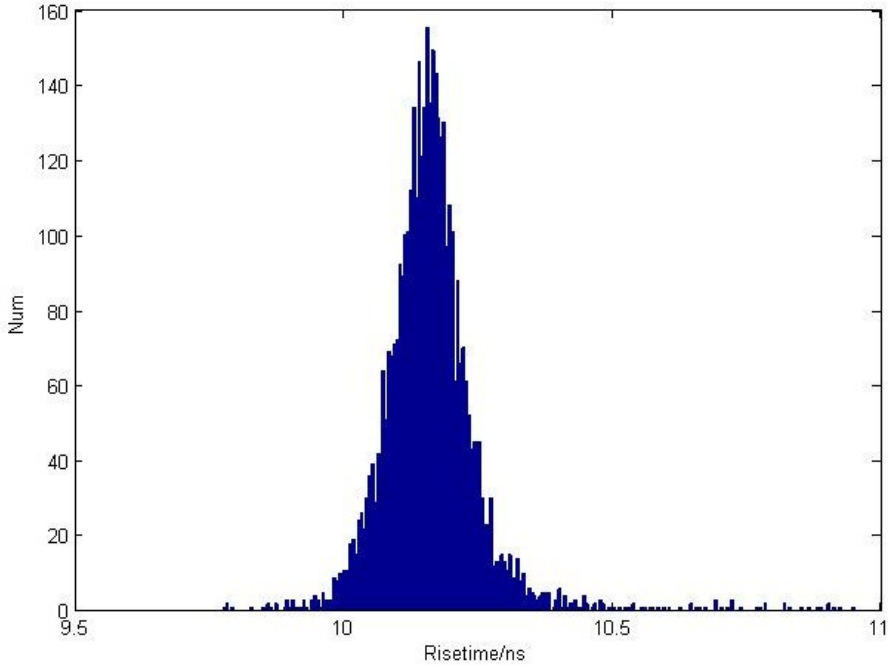


Figure 3.16 Shaped pulse rise-time distribution

CHAPTER 4

CONCLUSION

The capture-gated method has been demonstrated to be an excellent neutron spectrometry method in many previous researches. In this thesis, we performed a research on the reliability of the boron-loaded scintillator and its pulse shape discrimination properties based on simulation using the Geant4 toolkit. We chose EJ339A as a sample boron-loaded scintillator.

The basic assumption used in the capture-gated method is that neutron absorption only happens after the incident neutron loses all of its kinetic energy. The recoil-proton pulse amplitude can only be used to estimate the incident neutron energy when this assumption is true. However, as demonstrated in this thesis, this assumption is not always accurate. The neutron energy while being captured can be several or tens of percent of its original energy. As a result, degradation of neutron energy spectrum resolution is inevitable in this case. According to Monte Carlo simulation using Geant4 toolkit, the energy discrepancy is correlated to the capture time – the time interval between the first scattering and the final capture event. Moreover, the relation between capture energy and capture time stays intact when the energy of incident neutron changes. In this way, one can correct the energy of neutron of all energy ranges based on the same relation. Then, correction was implemented based on the established relation between capture time and

capture neutron energy. The achieved improvement on the energy resolution was demonstrated. The pulse shape discrimination performance of the detector was also examined using Geant4 simulation. The tail-to-total ratio algorithm is the main algorithm that this thesis focused on. In this study, a mono-energy neutron beam and a mono-energy gamma ray are simulated, respectively, which were directed to bombard the detector model created using the Geant4 toolkit. The information of optical photons was collected to form an output electric pulse. The tail-to-total ratio was calculated for each pulse generated by a gamma or neutron. After drawing histograms for the ratios for neutron and gamma, respectively, we found a separation of the ratio distributions for the recoil-proton pulses and gamma pulsed. These results illustrated the good pulse shape discrimination property of EJ339A.

In future, more simulations and experiments will be conducted to benchmark our results, especially the pulse shape discrimination property under experimental conditions. Furthermore, the reliability of this correction theory in real-world measurements will also be evaluated. Correction algorithms with better performance will be investigated.

APPENDIX A

IMPORTANT GEANT4 CODE

1. DetectorConstruction class:

```
ExN06DetectorConstruction::ExN06DetectorConstruction()
{
 filename = "output.gt4";
 defineSurfaces();

}

//....oooOO0OOooo........oooOO0OOooo........oooOO0OOooo........oooOO0OOooo......

ExN06DetectorConstruction::~ExN06DetectorConstruction(){;}

//....oooOO0OOooo........oooOO0OOooo........oooOO0OOooo........oooOO0OOooo......

G4VPhysicalVolume* ExN06DetectorConstruction::Construct()
{

//       ------------- Materials -------------

  G4double a,  density,fractionmass;
  G4int nel;

  G4NistManager* man = G4NistManager::Instance();

  G4Element *C = man->FindOrBuildElement("C");
  G4Element *H = man->FindOrBuildElement("H");
  G4Element *Si = man->FindOrBuildElement("Si");
  G4Element *O = man->FindOrBuildElement("O");
  G4Element *Sb = man->FindOrBuildElement("Sb");
  G4Element *Rb = man->FindOrBuildElement("Rb");
  G4Element *Cs = man->FindOrBuildElement("Cs");
  G4Isotope* B10 = new G4Isotope("Boron10", 5, 10, a= 10.00*g/mole);
  G4Isotope* B11 = new G4Isotope("Boron11", 5, 11, a= 11.00*g/mole);
  G4Element* EnBoron = new G4Element("EnBoron", "En_B", 2);
  EnBoron->AddIsotope(B10, 0.90);
```

```
EnBoron->AddIsotope(B11, 0.10);


  //------------------------------
 // Aluminum
  G4Material* Aluminum = man->FindOrBuildMaterial("G4_Al");
  Aluminum->SetName("Aluminum");
  //------------------------------
   //---------------------------------------------
  // Polydimethylsiloxane (Grease)
  G4Material* Polydimethylsiloxane = new G4Material("Polydimethylsiloxane", 0.97*g/cm3, 4,
kStateLiquid);
  Polydimethylsiloxane->AddElement(Si, 1);
  Polydimethylsiloxane->AddElement(O, 1);
  Polydimethylsiloxane->AddElement(C, 2);
  Polydimethylsiloxane->AddElement(H, 6);
  G4MaterialPropertiesTable* polydimethylsiloxaneprop = new G4MaterialPropertiesTable();
  const G4int numentriespolydimethylsiloxane = 3;
  G4double polydimethylsiloxaneenergy[numentriespolydimethylsiloxane] = {1.2*eV, 3.1*eV,
6.5*eV};
  G4double polydimethylsiloxaneabsorp[numentriespolydimethylsiloxane] = {10.*cm, 10.*cm,
10.*cm};
  G4double polydimethylsiloxanerindex[numentriespolydimethylsiloxane] = {1.4, 1.4, 1.4};
  polydimethylsiloxaneprop->AddProperty("ABSLENGTH", polydimethylsiloxaneenergy,
polydimethylsiloxaneabsorp, numentriespolydimethylsiloxane);
  polydimethylsiloxaneprop->AddProperty("RINDEX", polydimethylsiloxaneenergy,
polydimethylsiloxanerindex, numentriespolydimethylsiloxane);
  Polydimethylsiloxane->SetMaterialPropertiesTable(polydimethylsiloxaneprop);
  //---------------------------------------------

  //------------------------------
  // Fused silica
  G4Material* FusedSilica = new G4Material("FusedSilica", 2.201*g/cm3, 2, kStateSolid);
  FusedSilica->AddElement(Si, 1);
  FusedSilica->AddElement(O, 2);
  G4MaterialPropertiesTable* fusedsilicaprop = new G4MaterialPropertiesTable();
  const G4int numentriesfusedsilica = 3;
  G4double fusedsilicaenergy[numentriesfusedsilica] = {1.2*eV, 3.1*eV, 6.5*eV};
  G4double fusedsilicaabsorp[numentriesfusedsilica] = {2.*m, 2.*m, 2.*m};
  G4double fusedsilicarindex[numentriesfusedsilica] = {1.56, 1.47, 1.45};
  fusedsilicaprop->AddProperty("ABSLENGTH", fusedsilicaenergy, fusedsilicaabsorp,
numentriesfusedsilica);
  fusedsilicaprop->AddProperty("RINDEX", fusedsilicaenergy, fusedsilicarindex,
numentriesfusedsilica);
  FusedSilica->SetMaterialPropertiesTable(fusedsilicaprop);
  //------------------------------

  //------------------------------
  // Bialkali Cathode (dummy)
  G4Material* BialkaliCathode = new G4Material("BialkaliCathode", 3*g/cm3, 3, kStateSolid);
  BialkaliCathode->AddElement(Sb, 1);
```

```
  BialkaliCathode->AddElement(Rb, 1);
  BialkaliCathode->AddElement(Cs, 1);
  G4MaterialPropertiesTable* bialkalicathodeprop = new G4MaterialPropertiesTable();

bialkalicathodeprop->AddProperty("RINDEX", fusedsilicaenergy, fusedsilicarindex,
numentriesfusedsilica); // use values from window to prevent refraction
  BialkaliCathode->SetMaterialPropertiesTable(bialkalicathodeprop);
  //-----------------------------

//-----------------------------
  //Scintillator Material
  const G4int numentriesnai = 11;
  G4double naienergies[numentriesnai] = { 2.48*eV, 2.58*eV, 2.69*eV, 2.75*eV, 2.82*eV,
2.85*eV, 2.88*eV, 2.92*eV, 2.94*eV, 2.95*eV, 3.10*eV }; //  max: 425 nm
  G4double naifastcomp[numentriesnai] = { 0.05,    0.18,    0.44,    0.53,    0.75,    0.85,    0.95,
1.0,    0.93,    0.88,    0.0 };
  G4double naislowcomp[numentriesnai] = { 0.05,    0.18,    0.44,    0.53,    0.75,    0.85,    0.95,
1.0,    0.93,    0.88,    0.0 };
  G4double nairindices[numentriesnai] = { 1.415,    1.415,    1.415,    1.415,    1.415,    1.415,
1.415,    1.415,    1.415,    1.415,    1.415};//
  G4double naiabsorptionlength[numentriesnai] = { 1.0*m, 1.0*m, 1.0*m, 1.0*m, 1.0*m, 1.0*m,
1.0*m, 1.0*m, 1.0*m, 1.0*m, 1.0*m }; // guessed

  G4Material* Scin = new G4Material("Scin", density= 0.93*g/cm3, nel=4);

  Scin->AddElement(EnBoron, fractionmass=5.12*perCent);
  Scin->AddElement(C, fractionmass=62.53*perCent);
  Scin->AddElement(O, fractionmass=23.30*perCent);
  Scin->AddElement(H, fractionmass=9.04*perCent);
  G4MaterialPropertiesTable* naiprop = new G4MaterialPropertiesTable();
  naiprop->AddProperty("FASTCOMPONENT", naienergies, naifastcomp, numentriesnai);
  naiprop->AddProperty("SLOWCOMPONENT", naienergies, naislowcomp, numentriesnai);
  naiprop->AddProperty("RINDEX",       naienergies, nairindices, numentriesnai);
  naiprop->AddProperty("ABSLENGTH",    naienergies, naiabsorptionlength,  numentriesnai);
  naiprop->AddConstProperty("SCINTILLATIONYIELD",10.0/keV); //from data sheet
  naiprop->AddConstProperty("RESOLUTIONSCALE",1.0);
  naiprop->AddConstProperty("FASTTIMECONSTANT",3.7*ns); // from data sheet
  naiprop->AddConstProperty("SLOWTIMECONSTANT",3.7*ns); // from data sheet
  //naiprop->AddConstProperty("SLOWSCINTILLATIONRISETIME",27.0*ns); // not sure
  naiprop->AddConstProperty("YIELDRATIO",0.65); //light output???
  Scin->SetMaterialPropertiesTable(naiprop);

  Scin->GetIonisation()->SetBirksConstant(0.126*mm/MeV);

  ScintMater  = Scin;

  //-----------------------------
  // Air
  G4Material* Air = man->FindOrBuildMaterial("G4_AIR");
  Air->SetName("Air");
  const G4int numentries = 2;
```

```
  G4double energies[numentries] = { 0.1*eV, 10.0*eV };
  G4double vacrindices[numentries] = { 1., 1. };
  G4double airabsorpti[numentries] = { 100.*m, 100.*m }; // avoid infinite light-paths
  G4MaterialPropertiesTable* airprop = new G4MaterialPropertiesTable();
  airprop->AddProperty("ABSLENGTH", energies, airabsorpti, numentries);
  airprop->AddProperty("RINDEX", energies, vacrindices, numentries);
  Air->SetMaterialPropertiesTable(airprop);

//
//       ------------ Volumes --------------

  fSourceDistance = 3.0*cm;
  fScintDiameter = 76.2*mm;
  fScintThickness = 76.2*mm;
  fPMTDiameter = 76.2*mm;
  fWindowThickness = 2.0*mm;
  fEnvelopeThickness = 2.0*mm;
  fGreaseThickness = 0.1*mm;
  fShellThickness = 0.5*mm;
  fAirGap = 0.2*mm;
  fWorldLength = 10.0*m;
  fPMTLength = 50.0*mm;
  fCathodeThickness = 1.0*mm;

  G4Box*             solidWorld;    // pointer to the solid envelope
  G4LogicalVolume*   logicWorld;    // pointer to the logical envelope
  G4VPhysicalVolume* physiWorld;    // pointer to the physical envelope


  G4double HalfWorldLength = 0.5*fWorldLength;
  solidWorld= new G4Box("sworld",HalfWorldLength,HalfWorldLength,HalfWorldLength);
  logicWorld= new G4LogicalVolume( solidWorld, G4Material::GetMaterial("Air"), "lWorld", 0,
0, 0);

   //  Must place the World Physical volume unrotated at (0,0,0).
  //
  physiWorld = new G4PVPlacement(0, // no rotation
                 G4ThreeVector(), // at (0,0,0)
                 logicWorld,      // its logical volume
                 "pWorld",        // its name
                 0,             // its mother  volume
                 false, // no boolean operations
                 0);            // copy number

  // Set user cuts to avoid deadlocks
  G4double maxStep = 10.0*m, maxLength = 10.0*m, maxTime = 100*ns, minEkin = 0.5*eV;
  logicWorld->SetUserLimits(new G4UserLimits(maxStep,maxLength,maxTime,minEkin));

  //------------------------------
  // Scintillator
  //------------------------------
```

```
  G4Tubs*    solidScint;// pointer to the solid Scintillator
  G4LogicalVolume* logicScint;  // pointer to the logical Scintillator
  G4VPhysicalVolume*physiScint; // pointer to the physical Scintillator

 G4ThreeVector positionScint = G4ThreeVector(0, 0,
fScintThickness/2.0+fShellThickness+fAirGap);

 solidScint = new G4Tubs("Liquid", 0.*cm, fScintDiameter/2.0, fScintThickness/2.0, 0.*deg,
360.*deg);
 logicScint = new G4LogicalVolume(solidScint,ScintMater,"Liquid",0,0,0);
 physiScint = new G4PVPlacement(0, // no rotation
                    positionScint,  // at (x,y,z)
                   logicScint,     // its logical volume
                   "Liquid",       // its name
                   logicWorld, // its mother  volume
                  false, // no boolean operations
                   0); // copy number


       //-----------------------------------------------
 // Grease
 //-----------------------------------------------

 G4Tubs*           solidGrease;
 G4LogicalVolume*   logicGrease;
 G4VPhysicalVolume* physiGrease;

 G4ThreeVector positionGrease =
G4ThreeVector(0,0,fScintThickness+fShellThickness+fAirGap+fGreaseThickness/2.0);

 solidGrease = new G4Tubs("sgrease", 0.*cm, fScintDiameter/2.0, fGreaseThickness/2.0, 0.*deg,
360.*deg);
 logicGrease = new G4LogicalVolume(solidGrease,
G4Material::GetMaterial("Polydimethylsiloxane"), "lGrease", 0,0,0);
 physiGrease = new G4PVPlacement(0, // no rotation
              positionGrease,  // at (x,y,z)
              logicGrease,     // its logical volume
              "pGrease",       // its name
              logicWorld,      // its mother  volume
              false,          // no boolean operations
              0);             // copy number

  //------------------------------
 // PMT Window / Envelope
 //------------------------------

 G4ThreeVector positionEnvelope = G4ThreeVector(0, 0,
fScintThickness+fShellThickness+fAirGap+fGreaseThickness+fWindowThickness/2.0);

 G4Tubs*           solidWindow;
```

```
  G4Tubs*           solidShaft;
  G4UnionSolid*     solidEnvelope;
  G4LogicalVolume*  logicEnvelope;
  G4VPhysicalVolume* physiEnvelope;

  solidWindow = new G4Tubs("swindow", 0.*cm, fPMTDiameter/2.0, fWindowThickness/2.0,
0.*deg, 360.*deg);
  solidShaft = new G4Tubs("sshaft", fPMTDiameter/2.0-fEnvelopeThickness, fPMTDiameter/2.0,
fPMTLength/2.0, 0.*deg, 360.*deg);
  solidEnvelope = new G4UnionSolid("senvelope",
                           solidWindow,
                           solidShaft,
                           new G4RotationMatrix,
                           G4ThreeVector(0, 0, (fPMTLength+fWindowThickness)/2.0)
                           );
  logicEnvelope = new G4LogicalVolume(solidEnvelope,
G4Material::GetMaterial("FusedSilica"), "lEnvelope", 0,0,0);
  physiEnvelope = new G4PVPlacement(0, // no rotation
                  positionEnvelope, // at (x,y,z)
                  logicEnvelope, // its logical volume
                 "pEnvelope",  // its name
                  logicWorld,      // its mother  volume
                   false, // no boolean operations
                   0);         // copy number

  //------------------------------
  // PMT Cathode
  //------------------------------

  G4ThreeVector positionCathode = G4ThreeVector(0, 0,
fScintThickness+fShellThickness+fAirGap+fGreaseThickness+fWindowThickness+fCathodeThi
ckness/2.0);

  G4Tubs*           solidCathode;  // pointer to the solid Cathode
  G4LogicalVolume*  logicCathode;  // pointer to the logical Cathode
  G4VPhysicalVolume* physiCathode;  // pointer to the physical Cathode

  solidCathode = new G4Tubs("scath", 0.*cm, fPMTDiameter/2.0-fEnvelopeThickness,
fCathodeThickness/2.0, 0.*deg, 360.*deg);
  logicCathode = new G4LogicalVolume(solidCathode,
G4Material::GetMaterial("BialkaliCathode"), "lCath", 0,0,0);
  physiCathode = new G4PVPlacement(0, // no rotation
                    positionCathode,// at (x,y,z)
                    logicCathode,// its logical volume
                   "pCath",     // its name
                    logicWorld, // its mother  volume
                    false, // no boolean operations
                    0); // copy number


//------------------------------
```

```
  // Shell / Cap
  //------------------------------

  G4ThreeVector positionShell = G4ThreeVector(0, 0, fShellThickness/2.0); // boolean solid
positioned by cap's center

  G4Tubs*           solidShellShaft;  // pointer to the conic solid Shell
  G4Tubs*           solidShellCap;  // pointer to the solid Cap
  G4BooleanSolid*     solidShell; // pointer to the ellipsoidal solid Shell
  G4LogicalVolume*    logicShell;  // pointer to the logical Shell
  G4VPhysicalVolume*  physiShell;  // pointer to the physical Shell

  G4double shellLength = fScintThickness + fShellThickness + fAirGap;

  solidShellShaft = new G4Tubs("sconicalshell", fScintDiameter/2.0+fAirGap,
fScintDiameter/2.0+fShellThickness+fAirGap, shellLength/2.0, 0.*deg, 360.*deg);
  solidShellCap = new G4Tubs("scap", 0.*cm, fScintDiameter/2.0+fAirGap, fShellThickness/2.0,
0.*deg, 360.*deg);
  solidShell = new G4UnionSolid("sshell",
                          solidShellCap,
                          solidShellShaft,
                          new G4RotationMatrix,
                          G4ThreeVector(0, 0, (shellLength-fShellThickness)/2.0));
  logicShell = new G4LogicalVolume(solidShell, G4Material::GetMaterial("Aluminum"),
"lShell", 0,0,0);
  physiShell = new G4PVPlacement(0, positionShell, logicShell, "pShell", logicWorld, false, 0);

 //----------------------------
 // Air gap
 //----------------------------

  G4ThreeVector positionCap = G4ThreeVector(0, 0, fShellThickness+fAirGap/2.0); // boolean
solid positioned by cap's center

  G4Tubs*           solidAirCap;  // pointer to the solid Air between Scint and Cap
  G4LogicalVolume*    logicAirCap;  // pointer to the logical Air between Scint and Cap
  G4VPhysicalVolume*  physiAirCap;  // pointer to the physical Air between Scint and Cap

  solidAirCap = new G4Tubs("saircap", 0.*cm, fScintDiameter/2.0+fAirGap, fAirGap/2.0, 0.*deg,
360.*deg);
  logicAirCap = new G4LogicalVolume(solidAirCap, G4Material::GetMaterial("Air"), "lAirCap",
0,0,0);
  physiAirCap = new G4PVPlacement(0, positionCap, logicAirCap, "pAirCap", logicWorld, false,
0);


 // --------------------------------
 // surface
 // --------------------------------

  // setup surfaces
```

```
  G4LogicalBorderSurface* scint2AirFaceBorderSurface = 0;
  scint2AirFaceBorderSurface = new G4LogicalBorderSurface("scint2AirFaceBorderSurface",
physiScint, physiAirCap, polishedWhitePainted);

  G4LogicalBorderSurface* scint2AirLateralBorderSurface = 0;
  scint2AirLateralBorderSurface = new
G4LogicalBorderSurface("scint2AirLateralBorderSurface", physiScint, physiWorld,
groundWhitePainted);

  // shell / cap (not configureable)

  G4LogicalBorderSurface* airCap2ShellSurface = 0;
  airCap2ShellSurface = new G4LogicalBorderSurface("air2CapSurface1", physiAirCap,
physiShell, airGroundAluminum);

  G4LogicalBorderSurface* air2ShellSurface = 0;
  air2ShellSurface = new G4LogicalBorderSurface("air2ShellSurface", physiWorld, physiShell,
airGroundAluminum);

  G4LogicalBorderSurface* envelope2CathodeSurface = 0;
  envelope2CathodeSurface = new G4LogicalBorderSurface("envelope2CathodeSurface",
physiEnvelope, physiCathode, silicaCathodeMaterial);

 //------------------------------------------------
  // Define sensitive detector
  //------------------------------------------------

  G4SDManager* SDman = G4SDManager::GetSDMpointer();
  G4String sensitiveDetectorName = "/detector/sensitiveDetector";
  theCathodeSD = new CathodeSD(sensitiveDetectorName, physiCathode);
  SDman->AddNewDetector( theCathodeSD );
  logicEnvelope->SetSensitiveDetector( theCathodeSD );

return physiWorld;
}
```

PrimaryGeneratorAction class:

```
ExN06PrimaryGeneratorAction::ExN06PrimaryGeneratorAction()
{
  G4int n_particle = 1;
  particleGun = new G4ParticleGun(n_particle);

  //create a messenger for this class
  gunMessenger = new ExN06PrimaryGeneratorMessenger(this);

  //default kinematic
  //
```

```
  G4ParticleTable* particleTable = G4ParticleTable::GetParticleTable();
  G4ParticleDefinition* particle = particleTable->FindParticle("neutron");

  particleGun->SetParticleDefinition(particle);
  particleGun->SetParticleTime(0.0*ns);
  particleGun->SetParticlePosition(G4ThreeVector(0.0*cm,0.0*cm,-5.0*cm));
  particleGun->SetParticleMomentumDirection(G4ThreeVector(0.,0.,1.));
  particleGun->SetParticleEnergy(1000* keV);
}
```

2. SteppingAction class:

```
//....oooOO0OOooo.....oooOO0OOooo.....oooOO0OOooo.....oooOO0OOooo.....

ExN06SteppingAction::ExN06SteppingAction()
{
  fp=fopen("neutron.gt4","a");
       fp2=fopen("proton.gt4","a");
}

//....oooOO0OOooo.....oooOO0OOooo.....oooOO0OOooo.....oooOO0OOooo.....

void ExN06SteppingAction::UserSteppingAction(const G4Step* aStep)
{
   G4ParticleDefinition *def = aStep->GetTrack()->GetDefinition();
        G4double energy =  aStep->GetTrack()->GetKineticEnergy() / eV;
   //G4String particleType = def->GetParticleType(); // particle type = nucleus for d, t, He3, alpha,
and heavier nuclei
        G4String particleName = def->GetParticleName(); // e.g. for alpha: the name = "alpha" and
type = "nucleus"

   G4StepPoint* prePoint = aStep->GetPreStepPoint();
   //G4StepPoint* postPoint = aStep->GetPostStepPoint();

        G4double pre_energy =  prePoint->GetKineticEnergy() / eV;


   //G4double posX = aStep->GetTrack()->GetPosition().x() / cm ;
   //G4double posY = aStep->GetTrack()->GetPosition().y() / cm ;
        //G4double posZ = aStep->GetTrack()->GetPosition().z() / cm ;

        G4int TrackID = aStep->GetTrack()->GetTrackID();
        G4int ParentID = aStep->GetTrack()->GetParentID();

        G4double G_time = aStep->GetTrack()->GetGlobalTime() / ns;
        //G4double L_time = aStep->GetTrack()->GetLocalTime() / ns;
        //G4double P_time = aStep->GetTrack()->GetProperTime() / ns;

        G4int EventID = G4RunManager::GetRunManager()->GetCurrentEvent()->GetEventID();


        G4VPhysicalVolume* pVolume = prePoint->GetPhysicalVolume();



    if (particleName == "opticalphoton")  {
      //fprintf(fp, "%s\t",particleName.c_str());
          //fprintf(fp, "%8.6f\t%8.6f\t%8.6f\t",posX,posY,posZ);
      fprintf(fp, "%8.6f\t",G_time);
          //fprintf(fp, "%8.6f\t",L_time);
```

```
            //fprintf(fp, "%8.6f\t",P_time);
            fprintf(fp, "%8.6d\t",ParentID);
            //fprintf(fp, "%8.6d\t",TrackID);
            fprintf(fp, "%8.6d\n",EventID);
        }

          if (pVolume != NULL){
            G4String volumeName = pVolume->GetName();
            // If it happens in the Liquid

            if (volumeName == "Liquid"){


                    if (particleName == "neutron" && energy == 0.0)          {
            //write the current energy position to a file

        //fprintf(fp, "%s\t",particleName.c_str());
            fprintf(fp, "%8.6f\t",pre_energy);
            fprintf(fp, "%8.6f\t",energyDep);
        fprintf(fp, "%8.6f\t",G_time);
            //fprintf(fp, "%8.6f\t",L_time);
            //fprintf(fp, "%8.6f\t",P_time);
            fprintf(fp, "%8.6d\t",ParentID);
            fprintf(fp, "%8.6d\t",TrackID);
            fprintf(fp, "%8.6d\n",EventID);
        }

            if (particleName == "proton" && ParentID == 1)          {
            //write the current energy position to a file

          //fprintf(fp, "%s\t",particleType.c_str());
            fprintf(fp2, "%8.6f\t",pre_energy);
            fprintf(fp2, "%8.6f\t",energyDep);
             fprintf(fp2, "%8.6f\t",G_time);
            //fprintf(fp, "%8.6f\t",L_time);
            //fprintf(fp, "%8.6f\t",P_time);
            fprintf(fp2, "%8.6d\t",ParentID);
            fprintf(fp2, "%8.6d\t",TrackID);
            fprintf(fp2, "%8.6d\n",EventID);

            }


        }
          }

}
```

REFERENCES

[1] A. E. Evans and L. V. East, “Evaluation of a Gridded 3He Spectrometer Tube for Safeguards and Other Nuclear Applications,” Los Alamos Scientific Laboratory, Los Alamos, CA Rep. LA-5291-PR, LS-UR-78-2562,17-19, Apr. 1973.

[2] R. G. Watt and et al., “A sensitive neutron spectrometer for the National Ignition Facility,” Rev. Sci. Instrum., vol. 72, no. 1, pp. 846, 2001.

[3] J. L. Dolan and et al., “The estimation of neutron energy spectra of nuclear materials by passive measurements for nuclear nonproliferation applications,” IEEE Nuclear Science Symposuim & Medical Imaging Conference, Knoxville, TN, 2010, pp. 119–124.

[4] B. Czirr and et al., "Capture-gated neutron spectrometry," Nucl. Instrum. Meth. Phys. Res. A, vol. 476, no. 1–2, pp. 309–312, 2002.

[5] J. H. Kim and et al., "Neutron spectrometer with a capture-gated liquid scintillator," J. Korean. Phys. Soc., Vol. 56, No. 1, pp. 34-41, 2010.

[6] D.M. Drake and et al., “New electronically black neutron detectors,” Nucl. Instr. and Meth. A, Vol. 247, no. 3, pp. 576-582, 1986.

[7] H.P. Chou and C.Y. Horng, “Integral test of a boron-10 loaded liquid scintillator for neutron detection,” Nucl. Instr. and Meth. A, Vol. 328, no. 3, pp. 522-525, 1993.

[8] S. D. Jastaniah and P. J. Sellin, “Digital techniques for n/γ pulse shape discrimination and capture-gated neutron spectroscopy using liquid scintillators,” Nucl. Instr. and Meth. A, Vol. 517, no. 1-3, pp. 202-210, 2004.

[9] Tatjana Jevremovic, “Neutron Physics,” in *Nuclear Principles In Engineering,* 2nd ed. New York: Springer Science+Business Media, LLC, 2009, p288.

[10] *McGraw-Hill Dictionary of Scientific & Technical Terms*, 6th ed, New York, NY: The McGraw-Hill Companies, Inc., 2003.

[11] Glenn F. Knoll, “Slow Neutron Detection Methods,” Radiation Detection And Measurement, 3rd ed. New York, NY: John Wiley & Sons, Inc., 2000, p506.

[12] Yoshida, Y. and et al., "A practical method for evaluating the neutron dose equivalent rate." J. Nucl. Sci. Technol. Vol. 3, no. 11, pp. 473–478, 1966.

[13] Hornyak, W. F. "A Fast Neutron Detector," Rev. Sci. Instrum., vol. 23, no. 3, pp. 264–267, 1952.

[14] L.C. Johnson and et al., "Neutron diagnostics for ITER," Rev. Sci. Instr., vol. 68, no. 1, pp. 569, 1997.

[15] T. Elevant and et al., "Proposed neutron diagnostics for Wendelstein 7-X stellarator," Rev. Sci. Instr., vol. 70, no. 1, pp. 1185, 1999.

[16] A.J. Peurrung, "Recent developments in neutron detection," Nucl. Instrum. Meth. Phys. Res. A, vol. 443, no. 2-3, pp. 400-415, 2000.

[17] Saddig Darwish Jastaniah, "Development of a capture-gated fast neutron detector with psd using dsp," Ph.D. dissertation, Sch. of Elect. Phys. Sci., Univ. of Surrey, Guildford, UK, pp. 1–161, 2003.

[18] Nicholas Tsoulfanidis, "Scintillation Detectors" in *Measurement and detection of radiation*, 3rd ed., DC: Taylor & Francis, pp. 218, 2010.

[19] A. Raviart and Y. Koechlin, "Analyse par echantillonnage sur photons individuels des liquides fluorescents dans le domaine de la sub-nanoseconde," Nucl. Instrum. Meth., vol. 29, no. 1, pp. 45, 1964.

[20] F. J. Lynch, "New Liquid Scintillators with Higher Speed and Efficiency," IEEE Trans Nucl. Sci. vol. 15, no. 3, pp. 102, 1968.

[21] M. Hisrschberg et al., "Precise measurement of Birks kB parameter in plastic scintillators," IEEE Trans. Nucl. Sci., vol. 39, no. 4, pp. 511, 1992.

[22] J. B. Birks, "Scintillations from Organic Crystals: Specific Fluorescence and Relative Response to Different Radiations," Proc. Phys. Soc. A, vol. 64, pp. 874, 1951.

[23] J. B. Birks. "The Theory and Practice of Scintillation Counting," London, Pergamon Press, 1964.

[24] H.F. Atwater, "Monte Carlo calculation of recoil spectra in 4He proportional counters," Nucl. Instrum. Meth., vol. 100, pp. 453, 1972.

[25] E. A. Kamykowskim, "Comparison of calculated and measured spectral response and intrinsic efficiency for a boron-loaded plastic neutron detector" Nucl. Instrum. Meth. A, vol. 317, no. 3, pp. 559, 1992.

[26] T. Aoyama et al, “Energy response of a full-energy-absorption neutron spectrometer using boron-loaded liquid scintillator BC-523,” Nucl. Instrum. Meth. A, vol. 333, no. 2-3, pp. 492, 1993.

[27] J. B. Czirr and G. L. Jensen, “A compact neutron coincidence spectrometer, its measured response functions and potential applications,” Nucl. Instrum. Meth. A, vol. 349, no. 2-3, pp. 532, 1994.

[28] G. Ranucci, “An analytical approach to the evaluation of the pulse shape discrimination properties of scintillators,” Nucl. Instrum. Meth. A, vol. 354, no. 2-3, pp. 389–399, 1995.

[29] Jastaniah, S. D. and Sellin, P. J. “Digital techniques for n/γ pulse shape discrimination and capture-gated neutron spectroscopy using liquid scintillators,” Nucl. Instr. & Meth. Phys. Res A, vol. 517, no. 1-3, pp. 202–210, 2004.

[30] Pozzi, S. A. “Measurements of Continuous-in-Energy Neutron Sources Using the BC-523A Capture-Gated Liquid Scintilator,” IEEE Nuclear Science Symposium Conference, Orlando, FL, 2009, pp. 940-943.

[31] Sara A. Pozzi and et al., "Analysis of the response of capture-gated organic scintillators, " IEEE Trans. Nucl. Sci., vol. 52, no. 6, pp. 3141-3146, 2005.

[32] S. Agostinelli and et al., “Geant4—a simulation toolkit,” Nucl. Instr. & Meth. Phys. Res A, vol. 506, no. 3, pp. 250–303, 2003.